INTERVIEW® 8000 TURBO Series
Quick Reference Guide to the C Library

ERRATA

951–B0430–01
Issue 7
September 1995

INTERVIEW® 8000 TURBO Series Quick Reference Guide to the C Library

General Signal Networks
13000 Midlantic Drive
Mount Laurel, NJ 08054

ERRATA

Quick Reference Guide to the C Library

© 1992,1995 General Signal Networks

Notice

This is Issue 7 with Addendum (Jan 96) of *INTERVIEW 8000 TURBO Series Quick Reference Guide to the C Library.* This issue contains listings of variables, routines, and structures in alphabetical order and by subject together with a listing of definitions for each.

The current manual was written to specifications for INTERVIEW 8000 *TURBO* Series software version 12.00 (corresponding to Issue 3 of the *INTERVIEW 8000 TURBO Series Technical Manual*, September 1995, with Nov 95 Addendum). Subsequent software revisions may not affect the accuracy of this manual. The Addendum adds the TIM Expansion Shelf routines and updates our Internet address.

General Signal Networks is a newly formed company by the merger of Telenex, Data Switch, and Tau−Tron. Telenex Corporation is now General Signal Networks − Mount Laurel; we retain the same personnel and high standards which have always been hallmarks of Telenex Corporation and AR Test Systems.

General Signal Networks − Mount Laurel reserves the right to improve this manual or the equipment it describes without prior notice; we make no representation or warranty, expressed or implied, concerning the use or application of the material in this manual..

Any duplication of the material in this manual in any form without written permission from General Signal Networks − Mount Laurel is strictly forbidden.

For technical information, programming assistance, error decoding, and repairs, contact the factory. General Signal Networks − Mount Laurel is located in Mount Laurel, New Jersey, approximately 15 miles east of Philadelphia, PA. Local customers should call (609) 234-7900; in the rest of the U.S., call (800) 222-0187. Our Internet access is *http://www.gsnetworks.com*.

Contents

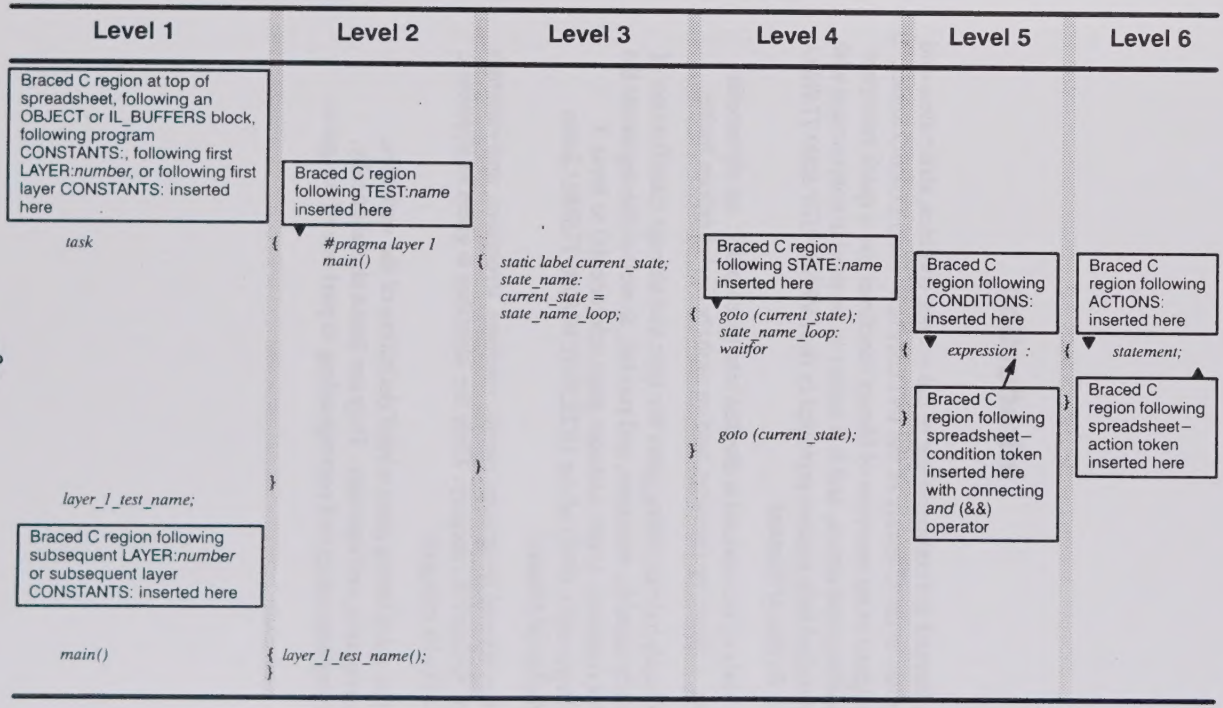
Introduction 1
Summary of C Regions 3
Pointer–List Buffer 5
Data–Character Buffer 7
Display Window 9
Trace Buffer 11
C Preprocessor *#pragmas* 13
C Library: Alphabetic Listing 15
C Library: Listing by Function 45
C Library: Alphabetic Description 57

Introduction

This manual is a compendium of the events, routines, structures, and variables in the C library of the INTERVIEW 8000 *TURBO* Series. It is designed as an overview of library functions and a quick reference for spelling and syntax, and it is meant to be used in conjunction with the detailed information provided in the *INTERVIEW 8000 TURBO Series Technical Manual.*

The body of the manual is divided into three parts: an alphabetic listing, a listing by function, and an alphabetic definitions listing.

- The alphabetic listing gives the type and storage classification of each variable, structure, and routine, as well as the argument list for routines. It also includes page reference(s) to Issue 3 (September 1995) of the *INTERVIEW 8000 TURBO Series Technical Manual.*
- The second list groups events, routines, structures, and variables by functional category. Only the identifier is given in alphabetic order by category.
- The third listing gives a brief definition of the variables, structures, and routines. They are listed alphabetically, complementing and corresponding to part one of the guide.


Level 1
Level 2
Level 3
Level 4
Level 5
Level 6
Braced C region at top of spreadsheet, following an OBJECT or IL_BUFFERS block, following program CONSTANTS:, following first LAYER:number, or following first layer CONSTANTS: inserted here
Braced C region following TEST:name inserted here
Braced C region following STATE:name inserted here
Braced C region following CONDITIONS: inserted here
Braced C region following ACTIONS: inserted here
task
{
#pragma layer 1
main()
{
static label current_state;
state_name:
current_state =
state_name_loop;
{
goto (current_state);
state_name_loop:
waitfor
{
expression :
{
statement;
Braced C region following spreadsheet–condition token inserted here with connecting and (&&) operator
Braced C region following spreadsheet–action token inserted here
goto (current_state);
}
}
}
}
layer_1_test_name;
Braced C region following subsequent LAYER:number or subsequent layer CONSTANTS: inserted here
main()
{ layer_1_test_name(); }

Summary of C Regions

The figure on the page opposite shows six levels of source code generated by the C translator when a program is compiled. Boxes at the various levels indicate the Protocol Spreadsheet "tokens"—OBJECT:, IL_BUFFERS:, CONSTANTS:, LAYER:, TEST:, STATE:, CONDITIONS:, ACTIONS:—that generated the source code. The boxes also point to the locations in the Spreadsheet program where braced C regions may be strategically placed.

Executable statements can occur at Levels 2, 3, 4, and 6 of the source code. Levels 1 and 2 are reserved for declarations.

The following rules apply to C declarations:

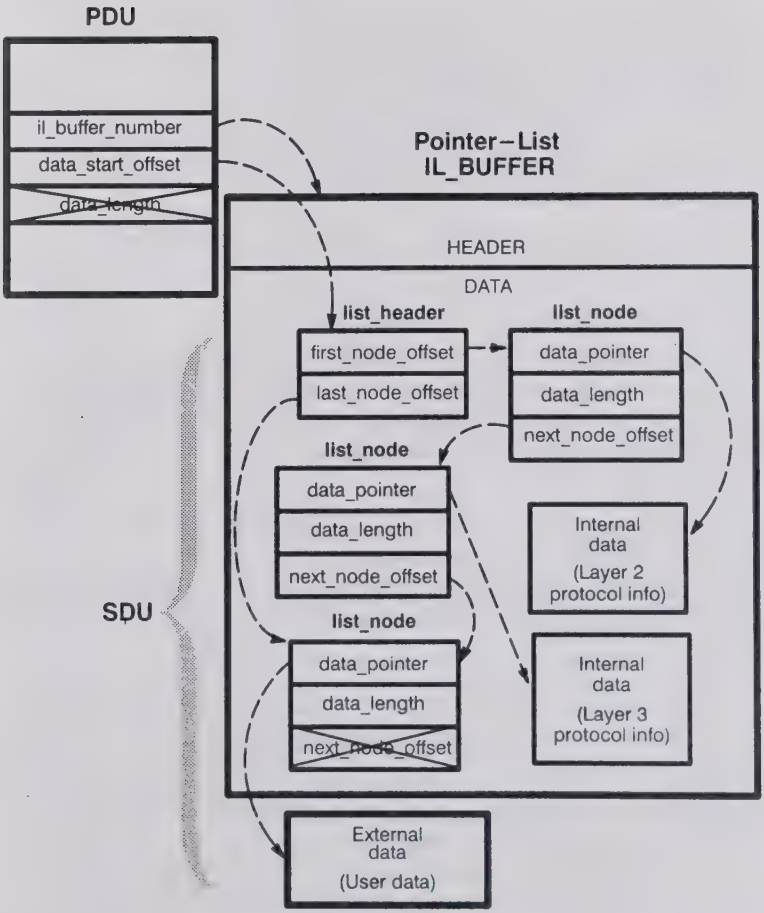
Declarations are legal following the left brace that introduces any compound statement. They should be grouped at the top of any region, since declarations are not allowed in a statement block below an executable statement. Declarations never are legal at Level 5; they are always legal at Level 1.

A variable or routine declared for Level 1 is declared for the remainder of Level 1 and across all levels to the right (the "telescoping" in the figure). A variable or routine declared at the top of Level 1 will be global throughout the program.

The set of variables listed as *extern* cannot be declared below Level 1.

A variable or routine may have its scope limited to a particular Test, State, or Actions block. It may also be redeclared at different levels.

A variable must be of the *static* storage class to pass its value into a *waitfor* statement. Declarations at Level 1 of the source code are always *static*, whether they are declared so explicitly or not. A variable that is initialized at Level 4 must be declared as *static* by the programmer if the initialized value is to be used inside a *waitfor*.


PDU
il_buffer_number
data_start_offset
data_length
Pointer–List
IL_BUFFER
HEADER
DATA
list_header
first_node_offset
last_node_offset
list_node
data_pointer
data_length
next_node_offset
list_node
data_pointer
data_length
next_node_offset
Internal
data
(Layer 2
protocol info)
SDU
list_node
data_pointer
data_length
next_node_offset
Internal
data
(Layer 3
protocol info)
External
data
(User data)

Pointer–List Buffer

The figure opposite shows a Primitive Data Unit and sample Pointer–List Buffer being passed down the layers.

IL Buffer Number. The buffer number to be passed with the primitive is always stored in the primitive. This buffer number is actually an 80286–processor segment number.

Data–Start Offset. In a pointer–list buffer going down the layers, the data–start offset will indicate the offset from the beginning of the buffer to the list–header node. This offset will vary if different linked lists have been started at different layers. Each list will have its own list header.

Data Length. In pointer–list buffers, the length of all data is unknown at any given layer.

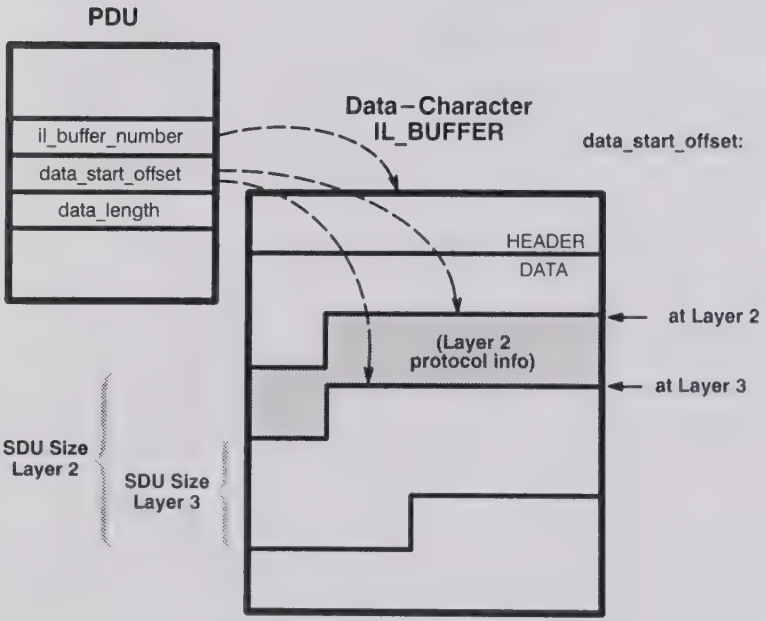
IL Buffer List Header. Each IL buffer contains a header that stores useful information, such as the status of the maintain bits that prevent the buffer from being returned to the general pool; the position of the buffered data in the INTERVIEW's display buffer; and the tick count (time) when the data was buffered from the line. (See *il_buffer* structure.)

Service Data Unit. The IL buffer also contains the data itself. This data component, the service data unit (or "SDU"), is added to as the buffer is passed down the layers.

The first part of the SDU in a pointer–list buffer is a *list–header node* (structure *il_list_header*) which contains information about the location of the first and last text nodes.

The SDU in a pointer–list buffer also includes *list nodes* (structure *il_list_node*) which give a pointer to data, the length of the data pointed to, and the offset from the start of the buffer to the *next* list node.

Finally, the service data unit in all buffers includes *data*, whether copied into the buffer (usually protocol information) or located in memory outside of the buffer (usually user data).


PDU
il_buffer_number
data_start_offset
data_length
Data–Character
IL_BUFFER
data_start_offset:
HEADER
DATA
at Layer 2
(Layer 2
protocol info)
at Layer 3
SDU Size
Layer 2
SDU Size
Layer 3

Data–Character Buffer

The figure opposite shows a Primitive Data Unit and sample Data–Character Buffer being passed up the layers.

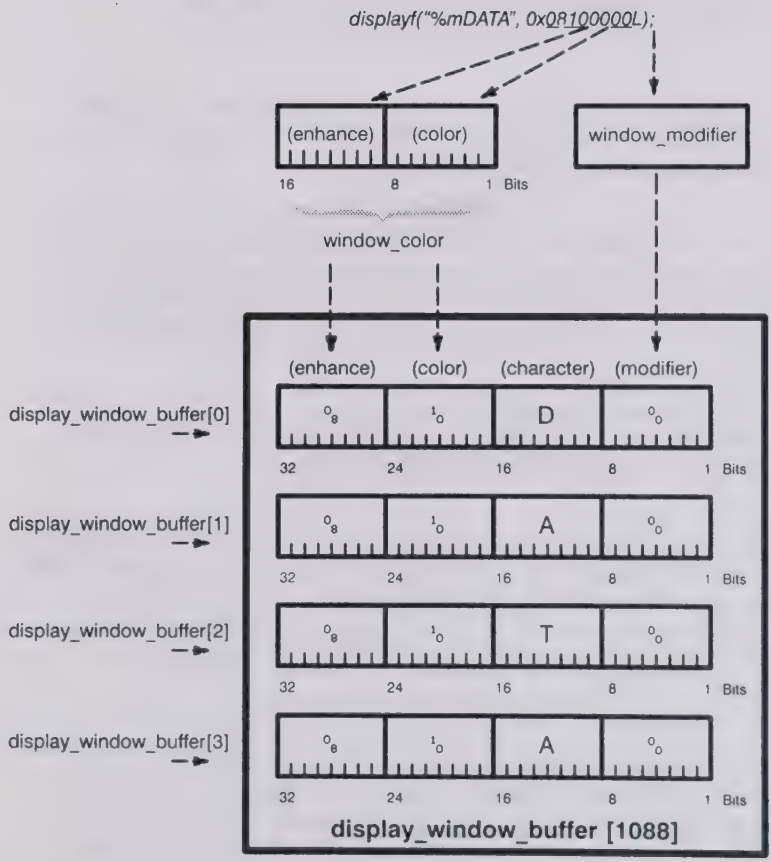
IL Buffer Number. The buffer number to be passed with the primitive is always stored in the primitive. This buffer number is actually an 80286–processor segment number.

Data–Start Offset. In a data–character buffer going up the layers, the data–start offset will change from layer to layer. For example, a buffer containing X.25 data that is being passed from Layer 2 to Layer 3 will have an offset at Layer 3 two bytes beyond the offset at Layer 2.

Data Length. In the example just given, the SDU will be smaller by two bytes at Layer 3 than it was at Layer 2.

IL Buffer List Header. Each IL buffer contains a header that stores useful information such as the status of the maintain bits that prevent the buffer from being returned to the general pool; the position of the buffered data in the INTERVIEW's display buffer; and the tick count (time) when the data was buffered from the line. (See *il_buffer* structure.)

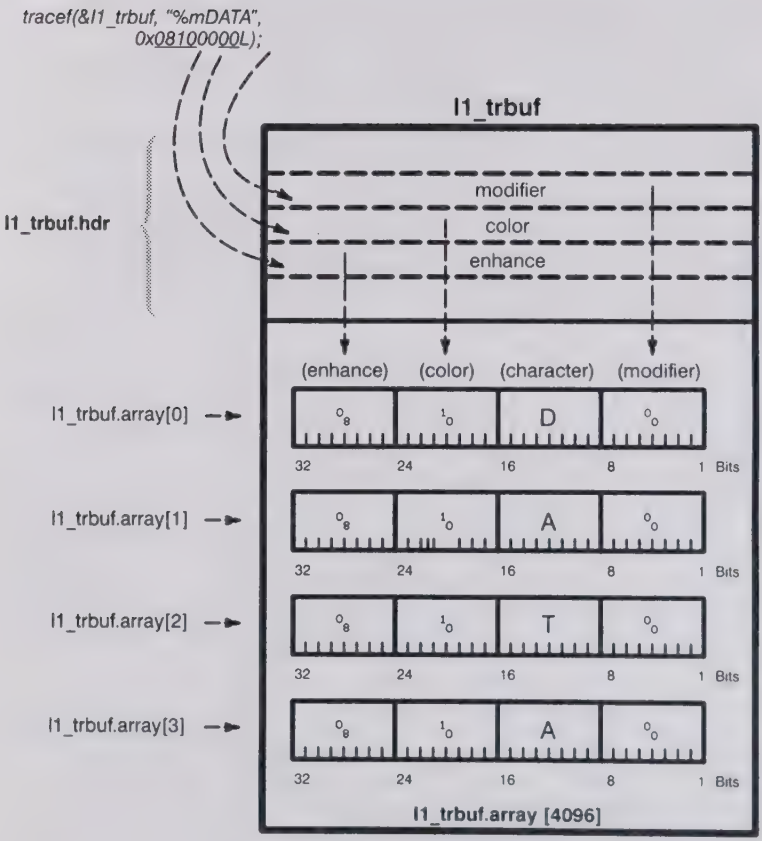
Service Data Unit. The IL buffer also contains the data itself. This data component, the service data unit (or "SDU"), is subtracted from as a buffer travels up the layers. *A data–character IL buffer includes all the data that was present when the data was first buffered*, and the contents of this buffer do not change as the buffer is passed up the layers. What changes is the service data unit, derived from the data–start offset in the PDU.


displayf("%mDATA", 0x0R100000L);
(enhance)
(color)
window_modifier
16
8
1 Bits
window_color
(enhance)
(color)
(character)
(modifier)
display_window_buffer[0]
display_window_buffer[1]
display_window_buffer[2]
display_window_buffer[3]
D
A
T
A
32
24
16
8
1 Bits
display_window_buffer [1088]

Display Window

When a *displayf* routine is called, the attributes assigned via the *%m* conversion specifier are stored in two *extern* variables, accessible to the user. Both color and enhance attributes are contained in *window_color*. The low byte in *window_color* indicates the color; the high byte contains enhancements.

In this example in the figure opposite, the following attributes will be assigned to characters written to the Display Window: reverse–image enhancement, green–on–black color, and ASCII font. Before a character is written to the Display Window, it is combined in a *long* with its attributes, as mapped in the figure.


tracef(&I1_trbuf, "%mDATA",
0x08100000L);
I1_trbuf
I1_trbuf.hdr
modifier
color
enhance
(enhance)
(color)
(character)
(modifier)
I1_trbuf.array[0]
I1_trbuf.array[1]
I1_trbuf.array[2]
I1_trbuf.array[3]
D
A
T
A
32
24
16
8
1 Bits
I1_trbuf.array [4096]

Trace Buffer

When a *tracef* routine is called, the attributes assigned via the *%m* conversion specifier are stored in three variables in the trace–buffer header of a designated buffer.

In the example on the page opposite, *l1_trbuf.hdr* holds the following attributes: reverse–image enhancement, green–on–black color, and ASCII font. Before a character is written to the buffer, it is combined in a *long* with its attributes, as mapped in the figure.

C Preprocessor *#pragmas*

The INTERVIEW 8000 *TURBO* Series supports preprocessor directives, including the full set of ANSI preprocessor directives. (Refer to the ANSI Recommendations for a discussion of these directives.) Some of these are discussed in the *C Basics* section of the Technical Manual.

As the name implies, all preprocessor directives are processed before the program in which they appear is compiled. Preprocessor directives are easy to recognize, since they are always preceded by a pound sign (#). Spaces are significant to the meaning of the directives, since other delimiters are generally not used. Note also that a semicolon cannot be used to terminate a preprocessor directive. Instead, a directive is terminated by a hard Carriage Return or some indicator of line continuation.

Implementation–defined *#pragmas* are also preprocessor directives. AR–defined *#pragmas* are given in the table below.

Identifier	Directive	Page
#pragma frame_nodes n	number	**80–3**
		82–4
#pragma hook	hook_type hook_text	**55–11**
#pragma il_buffer_size	size	**62–3**
#pragma il_buffers	number	**62–3**
#pragma layer	layer_number	**52–1**
#pragma nowarn		**A4–1**
#pragma object	filename	**55–8**
#pragma quad_display	on/off	**60–4**
#pragma sample_mode	% excess IL buffers	**80–1**
#pragma set_chain_xmit	on/off	**80–3**
#pragma set_high_speed_xmit	port_number on/off	**80–4**
#pragma smds_format E1		**82–4**
		83–5
#pragma smds_format DS1		**82–4**
		83–5
#pragma suppress_char_data		**58–7**
#pragma tracebuf	trace_number size	**60–31**

C Library: Alphabetic Listing

(Page numbers refer to the *INTERVIEW 8000 TURBO Series Technical Manual*, Issue 3, September 1995)

Identifier	Declaration	Page
	– A –	
a_conds	extern unsigned short	62–35
accumulator_struct	struct	61–14
last	unsigned long	
maximum	unsigned long	
minimum	unsigned long	
sample_count	unsigned short	
total_high	unsigned long	
total_low_low	unsigned short	
total_low_high	unsigned short	
changed	unsigned short	
activate_link	extern int	81–4
add_array_to_buff(array_ptr, count)	extern unsigned int	58–20
add_event_to_buff(event_word)	extern unsigned int	58–19
_append_il_buff_list_cnt(il_buffer_number, data_start_offset, text_ptr, text_length)	extern unsigned short	62–50
atm_cells_per_bundle	extern unsigned short	83–4
atm_framing_mode	extern unsigned short	83–4
atm_il_buffer	struct	83–1
lock	unsigned short	
maintain_bits	unsigned short	
buffer_size	unsigned short	
smds_resync_count	unsigned short	
receive_tag	unsigned short	
cell_busy_mask	unsigned short	
cell_crc10_error_mask	unsigned short	
atm_path_mask_1	unsigned short	
cell_hec_error_mask	unsigned short	
tick_count_high	unsigned short	
tick_count_mid	unsigned short	
tick_count_low	unsigned short	
available_space_offset	unsigned short	
bytes_remaining	unsigned short	
atm_path_mask_2	unsigned short	
atm_path_mask_3	unsigned short	
data [1]	unsigned char	
atm_mask_1	unsigned long	83–4
atm_mask_2	unsigned long	83–4
atm_mask_3	unsigned long	83–4
atm_path_1	unsigned long	83–4
atm_path_2	unsigned long	83–4
atm_path_3	unsigned long	83–4
atm_scrambling_selected	extern unsigned short	83–4
aux_change	extern fast_event	67–4
	– B –	
bcc_error	extern event	70–3 72–4 76–3
bcopy	extern void	68–20

Identifier	Declaration	Page
	– C –	
_chain_xmit [2]	extern unsigned char	80–3
clearerr(stream_ptr)	extern void	64–10
compute_actual_length	extern unsigned char	80–2
convert_tick_count(mpm_format_ticks, converted_ticks_ptr)	extern void	61–18
*counter*_name_*change*	extern event	61–3
counter_struct	struct	61–2
current	unsigned long	
last	unsigned long	
maximum	unsigned long	
minimum	unsigned long	
sample_count	unsigned short	
total_high	unsigned long	
total_low_low	unsigned short	
total_low_high	unsigned short	
prev	unsigned long	
old	unsigned long	
crnt_date_of_day	extern volatile const unsigned char	68–4
crnt_display_screen	extern unsigned short	60–2
crnt_time_of_day	extern volatile unsigned short	68–4
crnt_tm	extern struct tm	68–2
ctl_capture_d_rd(status)	extern void	75–6
ctl_capture_d_td(status)	extern void	75–7
ctl_capture_rd(status)	extern void	58–10
ctl_capture_td(status)	extern void	58–9
ctl_eia(on_mask, off_mask)	extern void	59–3 69–4
ctl_enhance_rd(enhance_type_status)	extern void	58–8
ctl_enhance_td(enhance_type_status)	extern void	58–7
curr_aux_value	extern volatile const unsigned short	67–4
current_col	extern unsigned short	60–6
current_eia_leads	extern const volatile unsigned short	59–2 69–3
current_line	extern unsigned short	60–6
	– D –	
d_channel_disabled	extern unsigned char	75–3
d_dce_frame	extern event	75–3
d_dce_frames_sup_run	extern unsigned long	78–7
d_dce_frames_suppressed	extern unsigned long	78–6
d_dte_frame	extern event	75–3
d_dte_frames_sup_run	extern unsigned long	78–6
d_dte_frames_suppressed	extern unsigned long	78–6
d_rcv_frame	extern event	75–3

Identifier	Declaration	Page
dce_abort	extern event	70–3 72–4 76–3 78–2
dce_bad_bcc	extern event	70–3 72–4 76–3 78–2
dce_flags	extern unsigned short	78–7
dce_frame	extern event	70–3 72–4 76–3 78–2
dce_frames_sup_run	extern unsigned long	78–6
dce_frames_suppressed	extern unsigned long	78–6
dce_good_bcc	extern event	70–3 72–4 76–3 78–2
dce_packet	extern event	71–3 77–2 79–2
deactivate_link	extern int	81–5
disable_dce( )	extern void	58–11
disable_dte( )	extern void	58–12
disable_freeze_disp_window( )	extern void	60–29
discard_d_dce_frames	extern unsigned char	80–2
discard_d_dte_frames	extern unsigned char	80–2
discard_dce_frames	extern unsigned char	80–2
discard_dte_frames	extern unsigned char	80–2
display_binary(hex_value)	void	57–5
display_prompt(string_ptr)	extern void	60–22
display_screen_changed	extern fast_event	60–2
display_window_buffer [1088]	extern unsigned long	60–8
display_window_index_buffer [17] mpm cpm	extern struct unsigned char unsigned char	60–9
displayc(character)	extern void	60–10
displayf(format_ptr, . . .)	extern int	60–10
displays(string_ptr)	extern void	60–21
dl_conds	extern unsigned short	62–15
dte_abort	extern event	70–3 72–4 76–3 78–2

Identifier	Declaration	Page
dte_bad_bcc	extern event	70–3 72–4 76–3 78–2
dte_flags	extern unsigned short	78–7
dte_frame	extern event	70–3 72–4 76–3 78–2
dte_frames_sup_run	extern unsigned long	78–6
dte_frames_suppressed	extern unsigned long	78–6
dte_good_bcc	extern event	70–3 72–4 76–3 78–2
dte_packet	extern event	71–3 77–2 79–2
_dup_il_buff_list(il_buffer_number, start_offset, new_start_offset_ptr)	extern unsigned short	62–41
_dup_il_buff_list_start(il_buffer_number, start_offset, new_start_offset_ptr)	extern unsigned short	62–39
	– E –	
enable_freeze_disp_window( )	extern void	60–29
enter_call_phase( )	extern void	69–10
enter_data_phase( )	extern void	69–11
eoc_message	struct	81–1
spare	unsigned int	
header	unsigned int	
message_type	unsigned int	
message_data	unsigned int	
evar_abort_d_rd	extern event	75–4
evar_abort_d_td	extern event	75–4
evar_bad_bcc_d_rd	extern event	75–4
evar_bad_bcc_d_td	extern event	75–4
evar_flag_d_rd	extern event	75–5
evar_flag_d_td	extern event	75–5
evar_gd_bcc_d_rd	extern event	75–5
evar_gd_bcc_d_td	extern event	75–5
evar_rcvd_char_d_rd	extern event	75–5
evar_rcvd_char_d_td	extern event	75–5
	– F –	
fclose(stream_ptr)	extern int	64–5
feof(stream_ptr)	extern int	64–7
ferror(stream_ptr)	extern int	64–8

Identifier	Declaration	Page
fevar_abort_d_rd	extern fast_event	75–4
fevar_abort_d_td	extern fast_event	75–4
fevar_abort_rd	extern fast_event	58–3
fevar_abort_td	extern fast_event	58–3
fevar_bd_bcc_d_rd	extern fast_event	75–4
fevar_bd_bcc_d_td	extern fast_event	75–5
fevar_bd_bcc_rd	extern fast_event	58–3 74–2
fevar_bd_bcc_td	extern fast_event	58–3 74–2
fevar_bd_bcc2_rd	extern fast_event	74–2
fevar_bd_bcc2_td	extern fast_event	74–2
fevar_d_dce_frame	extern fast_event	75–3
fevar_d_dte_frame	extern fast_event	75–3
fevar_d_rcv_frame	extern fast_event	75–3
fevar_d_xmit_cmplt	extern fast_event	75–3
fevar_eia_changed	extern fast_event	59–2 69–3
fevar_flag_d_rd	extern fast_event	75–5
fevar_flag_d_td	extern fast_event	75–5
fevar_flag_rd	extern fast_event	58–4
fevar_flag_td	extern fast_event	58–4
fevar_frm_error_rd	extern fast_event	58–3
fevar_frm_error_td	extern fast_event	58–4
fevar_gd_bcc_d_rd	extern fast_event	75–5
fevar_gd_bcc_d_td	extern fast_event	75–5
fevar_gd_bcc_rd	extern fast_event	58–3 74–2
fevar_gd_bcc_td	extern fast_event	58–3 74–2
fevar_gd_bcc2_rd	extern fast_event	74–2
fevar_gd_bcc2_td	extern fast_event	74–2
fevar_lo_dl_prmtv	extern fast_event	62–19
fevar_lo_n_prmtv	extern fast_event	62–23
fevar_lo_p_prmtv	extern fast_event	62–35
fevar_lo_ph_prmtv	extern fast_event	62–15
fevar_lo_s_prmtv	extern fast_event	62–31
fevar_lo_t_prmtv	extern fast_event	62–27
fevar_m_lo_dl_prmtv	extern fast_event	62–19
fevar_m_lo_n_prmtv	extern fast_event	62–23
fevar_m_lo_p_prmtv	extern fast_event	62–35
fevar_m_lo_ph_prmtv	extern fast_event	62–15
fevar_m_lo_s_prmtv	extern fast_event	62–31
fevar_m_lo_t_prmtv	extern fast_event	62–27

Identifier	Declaration	Page
fevar_parity_rd	extern fast_event	58–3
fevar_parity_td	extern fast_event	58–3
fevar_rcv_buffer_full	extern fast_event	58–4
fevar_rcvd_char_d_rd	extern fast_event	75–5
fevar_rcvd_char_d_td	extern fast_event	75–5
fevar_rcvd_char_rd	extern fast_event	58–3
fevar_rcvd_char_td	extern fast_event	58–3
fevar_time_of_day	extern fast_event	68–4
fevar_up_dl_prmtv	extern fast_event	62–15
fevar_up_n_prmtv	extern fast_event	62–19
fevar_up_p_prmtv	extern fast_event	62–31
fevar_up_s_prmtv	extern fast_event	62–27
fevar_up_t_prmtv	extern fast_event	62–23
fevar_xmit_cmplt	extern fast_event	58–4
fflush(stream_ptr)	extern int	64–6
fgetc(stream_ptr)	extern int	64–19
fgets(string_ptr, max_number, stream_ptr)	extern char *	64–18
*flag_*name*_change*	extern event	68–4
flag_struct	struct	68–2
prev	unsigned short	
current	unsigned short	
old	unsigned short	
flags_d_rd	extern unsigned long	78–7
flags_d_rd_max	extern unsigned short	78–8
flags_d_rd_min	extern unsigned short	78–9
flags_d_rd_run	extern unsigned long	78–8
flags_d_td	extern unsigned long	78–7
flags_d_td_max	extern unsigned short	78–8
flags_d_td_min	extern unsigned short	78–9
flags_d_td_run	extern unsigned long	78–8
flags_rd	extern unsigned long	78–7
flags_rd_max	extern unsigned short	78–8
flags_rd_min	extern unsigned short	78–9
flags_rd_run	extern unsigned long	78–8
flags_td	extern unsigned long	78–7
flags_td_max	extern unsigned short	78–8
flags_td_min	extern unsigned short	78–8
flags_td_run	extern unsigned long	78–7
fopen(filename_ptr, mode_ptr)	extern FILE *	64–4
fprintf(stream_ptr, format_ptr, . . .)	extern int	64–28
fputc(character, stream_ptr)	extern int	64–26
fputs(string_ptr, stream_ptr)	extern int	64–25
frame_disable_dce_1	void	80–7 82–7 83–8

Identifier	Declaration	Page
frame_disable_dce_2	void	80–7 82–7 83–8
frame_disable_dte_1	void	80–6 82–6 83–7
frame_disable_dte_2	void	80–8 82–8 83–9
frame_sent	extern event	70–3 72–5 76–3
fread(data_ptr, size, number, stream_ptr)	extern size_t	64–16
_free_il_msg_buff(il_buffer_number, relay_baton)	extern void	62–43
fseek(stream_ptr, bytes, reference)	extern int	64–11
fwrite(output_ptr, size, number, stream_ptr)	extern size_t	64–23
– G –		
get_68k_phys_addr(variable_ptr)	extern unsigned long	61–14
get_eoc_data	extern int	81–6
_get_file_type(pathname_ptr, type_buff_ptr)	extern int	64–37
_get_il_msg_buff(buffer_number_ptr, maintain_bit_ptr)	extern void	62–37
get_tim_setup	extern void	81–2
*get_user_timestamp (timestamp_struct *)*	void	68–23
get_wall_time_286_ticks(ticks_286_format_ptr)	extern void	61–17
get_wall_time_ticks(ticks_68K_format_ptr)	extern void	61–16
get_xtim_config	extern int	81–18
– H –		
hec_coset_value	extern unsigned short	83–4
_high_speed_xmit [2]	extern unsigned char	80–3
highlight_dw_fkey_label(fkey)	extern void	60–27
– I –		
idle_action(character)	extern void	58–15

Identifier	Declaration	Page
il_buffer	struct	62–10
lock	unsigned short	
maintain_bits	unsigned short	
buffer_size	unsigned short	
transmit_tag	unsigned short	
receive_tag	unsigned short	
char_buff_frame_start	unsigned long	
char_buff_frame_end	unsigned long	
tick_count_high	unsigned short	
tick_count_mid	unsigned short	
tick_count_low	unsigned short	
available_space_offset	unsigned short	
bytes_remaining	unsigned short	
bcc_indicator	unsigned long	
data [4064] †	unsigned char	
il_list_header	struct	60–12
first_node_offset	unsigned short	
last_node_offset	unsigned short	
reserved	unsigned long	
il_list_node	struct	62–12
data_pointer	unsigned char *	
data_length	unsigned short	
next_node_offset	unsigned short	
index(string, character)	extern char *	68–11
init_for_ds1( )	extern void	82–4
		83–6
init_for_e1( )	extern void	82–5
		83–6
_insert_il_buff_list_cnt(il_buffer_number, data_start_offset, text_ptr, text_length)	extern unsigned short	62–46
invalid_frame	extern event	70–3
		72–4
		76–3
invalid_packet	extern event	71–3
	– K –	
keyboard	extern struct	68–2
value	char	
keyboard_any_key	extern volatile unsigned short	68–5
keyboard_new_any_key	extern fast_event	68–5
keyboard_new_key	extern fast_event	68–5
	– L –	
l1_il_transmit(il_buffer_number, relay_baton, data_start_offset, transmit_tag)	extern void	58–14
l1_tick_count	extern unsigned long	61–10
l1_transmit(count, struct_send_string_ptr, xmit_tag)	extern void	58–13

† *il_buffer* size default; user–changeable by *#pragma il_buffer_size*

Identifier	Declaration	Page
l1_trbuf	extern struct trace_buf	**60–33**
hdr	struct trace_buffer_header	
array [4096] †	unsigned long	
l2_current_window_edge	extern volatile unsigned short	**70–5**
		72–6
		76–5
l2_enhance	extern unsigned char	**70–5**
		72–7
		76–6
		78–3
l2_give_data( )	extern void	**70–9**
		72–10
		76–9
l2_lower_window_edge	extern volatile unsigned short	**70–5**
		72–6
		76–5
l2_path table[ ]	struct	**72–3**
flags	unsigned char	
number	unsigned char	
c_addr	unsigned	
window_start	unsigned	
window_end	unsigned	
lower window_edge	unsigned	
upper_window_edge	unsigned	
current_window_edge	unsigned	
resend_edge	unsigned	
last_rcvd_nr	unsigned	
last_rcvd_ns	unsigned	
last_sent_nr	unsigned	
last_sent_ns	unsigned	
vr	unsigned	
vs	unsigned	
unack_frame_start	unsigned	
spare	unsigned	
l2_resend_edge	extern volatile unsigned short	**70–5**
		72–6
		76–5
l2_suppress	extern unsigned char	**70–5**
		72–7
		76–6
		78–3
l2_T1	extern event	**70–3**
		72–4
		76–3
l2_tick_count	extern unsigned long	**62–18**
l2_trbuf	extern struct trace_buf	**60–33**
hdr	struct trace_buffer_header	
array [4096] †	unsigned long	

† *il_buffer* size default; user–changeable by *#pragma il_buffer_size*

Identifier	Declaration	Page
l2_upper_window_edge	extern volatile unsigned short	70–5 72–6 76–5
l2pp_trbuff	extern unsigned char	60–48
l2pp_trbuff_ctl	extern struct lpp_trbuff_ctl	60–50
l2pp_trbuff_end	extern unsigned long	60–48
l3_clear_path(path_number)	extern void	71–11
l3_enhance	extern unsigned char	71–7 77–3 79–6
l3_give_data( )	extern void	71–10
l3_more_to_resend(path_number)	extern unsigned char	71–11
l3_suppress	extern unsigned char	71–7 77–3 79–6
l3_tick_count	extern unsigned long	62–22
l3_trbuf hdr array [4096] †	extern struct trace_buf struct trace_buffer_header unsigned long	60–33
l3_window_empty(path_number)	extern unsigned char	71–13
l3_window_full(path_number)	extern unsigned char	71–12
l3pp_trbuff	extern unsigned char	60–48
l3pp_trbuff_ctl	extern struct lpp_trbuff_ctl	60–51
l3pp_trbuff_end	extern unsigned long	60–48
l4_tick_count	extern unsigned long	62–26
l4_trbuf hdr array [4096] †	extern struct trace_buf struct trace_buffer_header unsigned long	60–33
l5_tick_count	extern unsigned long	62–30
l5_trbuf hdr array [4096] †	extern struct trace_buf struct trace_buffer_header unsigned long	60–34
l6_tick_count	extern unsigned long	62–34
l6_trbuf hdr array [4096] †	extern struct trace_buf struct trace_buffer_header unsigned long	60–34
l7_tick_count	extern unsigned long	62–37
l7_trbuf hdr array [4096] †	extern struct trace_buf struct trace_buffer_header unsigned long	60–34
lo_dl_il_buff	extern volatile unsigned short	62–21
lo_dl_pdu_seg	extern volatile unsigned short	62–20
lo_dl_prmtv	extern event	62–19
lo_dl_prmtv_avail( )	extern unsigned short	62–66
lo_dl_prmtv_code	extern volatile const unsigned char	62–20

† *il_buffer* size default; user–changeable by *#pragma il_buffer_size*

Identifier	Declaration	Page
lo_dl_prmtv_path	extern volatile const unsigned char	62–20
lo_dl_sdu	extern volatile unsigned short	62–21
lo_n_il_buff	extern volatile unsigned short	62–25
lo_n_pdu_seg	extern volatile unsigned short	62–24
lo_n_prmtv	extern event	62–23
lo_n_prmtv_avail( )	extern unsigned short	62–73
lo_n_prmtv_code	extern volatile const unsigned char	62–24
lo_n_prmtv_path	extern volatile const unsigned char	62–24
lo_n_sdu	extern volatile unsigned short	62–25
lo_p_il_buff	extern volatile unsigned short	62–36
lo_p_pdu_seg	extern volatile unsigned short	62–35
lo_p_prmtv	extern event	62–35
lo_p_prmtv_avail( )	extern unsigned short	62–91
lo_p_prmtv_code	extern volatile const unsigned char	62–36
lo_p_prmtv_path	extern volatile const unsigned char	62–36
lo_p_sdu	extern volatile unsigned short	62–37
lo_ph_il_buff	extern volatile unsigned short	62–17
lo_ph_pdu_seg	extern volatile unsigned short	62–16
lo_ph_prmtv	extern event	62–15
lo_ph_prmtv_avail( )	extern unsigned short	62–58
lo_ph_prmtv_code	extern volatile const unsigned char	62–16
lo_ph_prmtv_path	extern volatile const unsigned char	62–16
lo_ph_sdu	extern volatile unsigned short	62–17
lo_s_il_buff	extern volatile unsigned short	62–33
lo_s_pdu_seg	extern volatile unsigned short	62–32
lo_s_prmtv	extern event	62–31
lo_s_prmtv_avail( )	extern unsigned short	62–87
lo_s_prmtv_code	extern volatile const unsigned char	62–32
lo_s_prmtv_path	extern volatile const unsigned char	62–32
lo_s_sdu	extern volatile unsigned short	62–33
lo_t_il_buff	extern volatile unsigned short	62–29
lo_t_pdu_seg	extern volatile unsigned short	62–28
lo_t_prmtv	extern event	62–27
lo_t_prmtv_avail( )	extern unsigned short	62–80
lo_t_prmtv_code	extern volatile const unsigned char	62–28
lo_t_prmtv_path	extern volatile const unsigned char	62–28
lo_t_sdu	extern volatile unsigned short	62–29
load_program(filename_ptr)	extern void	68–12
lock(lock_variable_ptr)	extern void	64–14 68–13

Identifier	Declaration	Page
lpp_trbuff_ctl	struct	60–50
begin_off_hi	unsigned char	
begin_off_lo	unsigned char	
begin_wrap_hi	unsigned char	
begin_wrap_lo	unsigned char	
end_off_hi	unsigned char	
end_off_lo	unsigned char	
end_wrap_hi	unsigned char	
end_wrap_lo	unsigned char	
	– M –	
m_a_conds	extern unsigned short	62–35
m_bib	extern volatile const unsigned char	78–2
m_call_ref_flag	extern volatile const unsigned char	77–2
m_call_ref_len	extern volatile const unsigned char	77–3
m_cic	extern volatile unsigned short	79–6
m_code_type	extern volatile const unsigned char	79–2
m_dl_conds	extern unsigned short	62–15
m_fib	extern volatile const unsigned char	78–2
m_frame_addr	extern volatile const unsigned char	70–4 72–5
m_frame_addr_cr	extern volatile const unsigned char	76–4
m_frame_addr_sapi	extern volatile const unsigned char	76–4
m_frame_addr_tei	extern volatile const unsigned char	76–4
m_frame_bcc_type	extern volatile const unsigned char	70–4 72–5 76–4 78–3
m_frame_cntrl_byte_1	extern volatile const unsigned char	70–4 72–5 76–4
m_frame_nr	extern volatile const unsigned char	76–4
m_frame_ns	extern volatile const unsigned char	76–4
m_frame_pf	extern volatile const unsigned char	70–4 72–5 76–4
m_frame_type	extern volatile const unsigned char	70–4 72–5 73–1 76–4
m_info_element_len	extern volatile const unsigned char	77–3
m_label_dpc	extern volatile unsigned long	79–6
m_label_opc	extern volatile unsigned long	79–6
m_label_sls	extern volatile const unsigned char	79–6
m_li	extern volatile const unsigned char	78–3
m_lo_dl_il_buff	extern volatile unsigned short	62–21
m_lo_dl_pdu_seg	extern volatile unsigned short	62–20

Identifier	Declaration	Page
m_lo_dl_prmtv	extern event	62–19
m_lo_dl_prmtv_avail( )	extern unsigned short	62–66
m_lo_dl_prmtv_code	extern volatile const unsigned char	62–20
m_lo_dl_prmtv_path	extern volatile const unsigned char	62–20
m_lo_dl_sdu_offset	extern volatile unsigned short	62–21
m_lo_dl_sdu_size	extern volatile unsigned short	62–21
m_lo_n_il_buff	extern volatile unsigned short	62–25
m_lo_n_pdu_seg	extern volatile unsigned short	62–24
m_lo_n_prmtv	extern event	62–23
m_lo_n_prmtv_avail( )	extern unsigned short	62–73
m_lo_n_prmtv_code	extern volatile const unsigned char	62–24
m_lo_n_prmtv_path	extern volatile const unsigned char	62–24
m_lo_n_sdu_offset	extern volatile unsigned short	62–25
m_lo_n_sdu_size	extern volatile unsigned short	62–25
m_lo_p_il_buff	extern volatile unsigned short	62–36
m_lo_p_pdu_seg	extern volatile unsigned short	62–36
m_lo_p_prmtv	extern event	62–35
m_lo_p_prmtv_avail( )	extern unsigned short	62–91
m_lo_p_prmtv_code	extern volatile const unsigned char	62–36
m_lo_p_prmtv_path	extern volatile const unsigned char	62–36
m_lo_p_sdu_offset	extern volatile unsigned short	62–37
m_lo_p_sdu_size	extern volatile unsigned short	62–37
m_lo_ph_il_buff	extern volatile unsigned short	62–17
m_lo_ph_pdu_seg	extern volatile unsigned short	62–16
m_lo_ph_prmtv	extern event	62–15
m_lo_ph_prmtv_avail( )	extern unsigned short	62–59
m_lo_ph_prmtv_code	extern volatile const unsigned char	62–16
m_lo_ph_prmtv_path	extern volatile const unsigned char	62–16
m_lo_ph_sdu_offset	extern volatile unsigned short	62–17
m_lo_ph_sdu_size	extern volatile unsigned short	62–17
m_lo_s_il_buff	extern volatile unsigned short	62–33
m_lo_s_pdu_seg	extern volatile unsigned short	62–32
m_lo_s_prmtv	extern event	62–31
m_lo_s_prmtv_avail( )	extern unsigned short	62–87
m_lo_s_prmtv_code	extern volatile const unsigned char	62–32
m_lo_s_prmtv_path	extern volatile const unsigned char	62–32
m_lo_s_sdu_offset	extern volatile unsigned short	62–33
m_lo_s_sdu_size	extern volatile unsigned short	62–33
m_lo_t_il_buff	extern volatile unsigned short	62–29
m_lo_t_pdu_seg	extern volatile unsigned short	62–28
m_lo_t_prmtv	extern event	62–27
m_lo_t_prmtv_avail( )	extern unsigned short	62–80
m_lo_t_prmtv_code	extern volatile const unsigned char	62–28

Identifier	Declaration	Page
m_lo_t_prmtv_path	extern volatile const unsigned char	62–28
m_lo_t_sdu_offset	extern volatile unsigned short	62–29
m_lo_t_sdu_size	extern volatile unsigned short	62–29
m_message_type	extern volatile const unsigned char	77–3
m_message_type_defined	extern volatile const unsigned char	77–2
m_n_conds	extern unsigned short	62–19
m_p_conds	extern unsigned short	62–31
m_packet_bcc_type	extern volatile const unsigned char	77–2
m_packet_buff_seg	extern volatile unsigned short	71–5
m_packet_cause	extern volatile const unsigned char	71–4
m_packet_d	extern volatile const unsigned char	71–3
m_packet_daf	extern volatile const unsigned short	73–2
m_packet_def	extern volatile const unsigned short	73–2
m_packet_diag_code	extern volatile const unsigned char	71–4
m_packet_dsaf	extern volatile const unsigned long	73–2
m_packet_fi	extern volatile const unsigned char	73–3
m_packet_fid_type	extern volatile const unsigned char	73–2
m_packet_info_length	extern volatile unsigned short	71–5
m_packet_info_offset	extern volatile unsigned short	71–5
m_packet_info_ptr	extern volatile const unsigned char *	71–6
m_packet_info_seg	extern volatile unsigned short	71–5
m_packet_lcn	extern volatile const unsigned short	71–3
m_packet_lcn_grp	extern volatile unsigned char	71–3
m_packet_length	extern volatile unsigned short	71–5 73–2
m_packet_lsid	extern volatile const unsigned char	73–2
m_packet_m	extern volatile const unsigned char	71–3
m_packet_oaf	extern volatile const unsigned short	73–2
m_packet_oef	extern volatile const unsigned short	73–2
m_packet_osaf	extern volatile const unsigned long	73–2
m_packet_pr	extern volatile const unsigned char	71–3
m_packet_ps	extern volatile const unsigned char	71–3
m_packet_ptr	extern volatile unsigned char *	71–6
m_packet_q	extern volatile const unsigned char	71–3
m_packet_rri	extern volatile const unsigned char	73–3
m_packet_rti	extern volatile unsigned char	73–3
m_packet_ru_category	extern volatile const unsigned char	73–3
m_packet_sdi	extern volatile unsigned char	73–3
m_packet_sdu_offset	extern volatile unsigned short	71–5
m_packet_type	extern volatile const unsigned char	71–4
m_packet_type_byte	extern volatile const unsigned char	71–4
m_prot_disc	extern volatile const unsigned char	77–2
m_ptr_to_call_ref	extern volatile const unsigned char *	77–3

Identifier	Declaration	Page
m_ptr_to_info_element	extern volatile const unsigned char *	77–3
m_s_conds	extern unsigned short	62–27
m_sio_ni	extern volatile const unsigned char	79–2
m_sio_priority	extern volatile const unsigned char	79–2
m_sio_si	extern volatile const unsigned char	79–2
m_so0	extern volatile const unsigned char	78–3
m_t_conds	extern unsigned short	62–23
m_unit_type	extern volatile const unsigned char	78–2
mkdir(directory_ptr)	extern int	64–33
mpm_info	extern struct	65–6
rev_num	unsigned char	
present	unsigned char	
lo_addr	unsigned long	
hi_addr	unsigned long	
– N –		
n_conds	extern unsigned short	62–19
nr_error	extern event	70–3 72–4 76–3
ns_error	extern event	70–3 72–4 76–3
– O –		
_open_space_in_il_buff(il_buffer_number, length, space_offset_ptr)	extern void	62–42
outsync_action(side)	extern void	58–11
– P –		
p_conds	extern unsigned short	62–31
packet_sent	extern event	71–3
pdu	struct	62–9
primitive_code	unsigned char	
path	unsigned char	
parameter	unsigned long	
relay_baton	unsigned short	
il_buffer_number	unsigned short	
buffer_contents	unsigned char	
reserved	unsigned char	
data_start_offset	unsigned short	
data_length	unsigned short	
ph_ch2_discard_ind	extern volatile unsigned char	62–14
ph_discard_ind	extern volatile unsigned char	62–14
ph_prmtv_type	extern volatile unsigned char	62–14
pos_cursor(row, column)	extern unsigned int	60–23
pr_error	extern event	71–3
#pragma	(see page 13 of this Guide for explanation)	

Identifier	Declaration	Page
prev_aux_value	extern volatile const unsigned short	67–4
prev_date_of_day	extern volatile const unsigned char	68–5
prev_display_screen	extern unsigned short	60–3
prev_time_of_day	extern volatile unsigned short	68–4
prev_tm	extern struct tm	68–2
previous_eia_leads	extern const volatile unsigned short	59–2 69–3
print_buffer	struct	63–2
in	unsigned short	
out	unsigned short	
buffer_end	unsigned short	
lock	unsigned short	
polling	char	
overrun	char	
buffer [8192]	char	
_print_buffer	extern struct print_buffer	63–2
printc(character)	extern void	63–4
printf(format_ptr, . . .)	extern int	63–4
prints(string_ptr)	extern void	63–10
prog_trbuf	extern struct trace_buf	60–33
hdr	struct trace_buffer_header	
array [4096]	unsigned long	
ps_error	extern event	71–3

– R –

_raw_disk_read (dev_num, block, count, buffer)	extern short	68–24
_raw_disk_write (dev_num, block, count, buffer)	extern short	68–26
rcv_buffer_full	extern volatile unsigned short	58–4
rcvd_char_d_rd	extern unsigned short	75–4
rcvd_char_d_td	extern unsigned short	75–4
rcvd_char_rd	extern unsigned short	58–4
rcvd_char_td	extern unsigned short	58–4
rcvd_device_path	extern volatile const unsigned char	71–7
rcvd_frame	extern event	70–3 72–4 76–3
rcvd_frame_addr	extern volatile const unsigned char	70–4 72–5
rcvd_frame_addr_cr	extern volatile const unsigned char	76–4
rcvd_frame_addr_sapi	extern volatile const unsigned char	76–4
rcvd_frame_addr_tei	extern volatile const unsigned char	76–4
rcvd_frame_bcc_type	extern volatile const unsigned char	70–4 72–6 76–5

Identifier	Declaration	Page
rcvd_frame_buff_seg	extern volatile unsigned short	70–5 72–6 76–5
rcvd_frame_cntrl_byte_1	extern volatile const unsigned char	70–4 72–5 76–5
rcvd_frame_nr	extern volatile const unsigned char	70–4 72–6 76–5
rcvd_frame_ns	extern volatile const unsigned char	70–5 72–6 76–5
rcvd_frame_pf	extern volatile const unsigned char	70–4 72–5 76–5
rcvd_frame_sdu_offset	extern volatile unsigned short	70–5 72–6 76–5
rcvd_frame_sdu_size	extern volatile unsigned short	70–5 72–6 76–5
rcvd_frame_type	extern volatile const unsigned char	70–4 72–5 76–4
rcvd_packet	extern event	71–3
rcvd_packet_ptr	extern volatile unsigned char *	71–6
rcvd_packet_type	extern volatile const unsigned char	71–5
rcvd_pkt_buff_seg	extern volatile unsigned short	71–6
rcvd_pkt_cause	extern volatile const unsigned char	71–4
rcvd_pkt_d	extern volatile const unsigned char	71–4
rcvd_pkt_diagn	extern volatile const unsigned char	71–4
rcvd_pkt_info_length	extern volatile unsigned short	71–6
rcvd_pkt_info_offset	extern volatile unsigned short	71–6
rcvd_pkt_info_ptr	extern volatile unsigned char *	71–6
rcvd_pkt_info_seg	extern volatile unsigned short	71–6
rcvd_pkt_lcn	extern volatile unsigned short	71–4
rcvd_pkt_length	extern unsigned short	71–6
rcvd_pkt_m	extern volatile const unsigned char	71–4
rcvd_pkt_pr	extern volatile const unsigned char	71–4
rcvd_pkt_ps	extern volatile const unsigned char	71–4
rcvd_pkt_q	extern volatile const unsigned char	71–4
rcvd_pkt_sdu_offset	extern unsigned short	71–6
rcvd_pkt_type_byte	extern volatile const unsigned char	71–5
rd_1_resyncs	extern unsigned short	83–4
rd_1_smds_out_frame	extern unsigned char	82–3 83–4

Identifier	Declaration	Page
rd_buffer_off	extern unsigned short	58–5 74–2
rd_il_buffer	extern unsigned short	58–5 74–2
rd_modifier	extern unsigned char	58–5
remove(file_ptr)	extern int	64–33
rename(oldfile_ptr, newfile_ptr)	extern int	64–32
resend_frame(pf, first_or_next)	extern void	70–10 72–11 76–10
resend_frame_multi(pf, first_or_next, addr_value)	extern void	72–12
resend_packet(path_number, first_or_next)	extern void	71–14
reset_nr( )	extern void	70–11 72–13 76–11
reset_nr_multi(addr_value)	extern void	72–14
reset_ns( )	extern void	70–11 72–15 76–12
reset_ns_multi(addr_value)	extern void	72–15
reset_pr_ps(path_number)	extern void	71–15
reset_print_page( )	extern int	63–10
restore_cursor(position)	extern void	60–24
rewind(stream_ptr)	extern void	64–13
rh_ptr	extern volatile unsigned char *	73–3
rindex(string, character)	extern char *	68–12
rmt_break	extern event	66–3
rmt_flushi( )	extern int	66–8
rmt_flusho( )	extern int	66–16
rmt_get_baud_rate( )	extern int	66–25
rmt_get_bits( )	extern int	66–26
rmt_get_mode( )	extern int	66–27
rmt_get_parity( )	extern int	66–26
rmt_getc(wait)	extern int	66–4
rmt_getl(string_ptr, max_length)	extern int	66–5
rmt_gets(string_ptr, length, wait)	extern int	66–6
rmt_input_almost_empty	extern event	66–3
rmt_input_almost_full	extern event	66–3
rmt_input_empty	extern event	66–3
rmt_input_not_empty	extern event	66–3
rmt_input_overflow	extern event	66–3
rmt_lock( )	extern void	66–9
rmt_output_empty	extern event	66–3

Identifier		Declaration	Page
rmt_putb(string_ptr, length, wait)		extern int	66–14
rmt_putc(character, wait)		extern int	66–11
rmt_puts(string_ptr, wait)		extern int	66–13
rmt_resumeo( )		extern int	66–19
rmt_send_break(wait)		extern int	66–19
rmt_set_baud_rate(speed)		extern int	66–21
rmt_set_bits(value)		extern int	66–22
rmt_set_mode(mode)		extern int	66–24
rmt_set_parity(parity)		extern int	66–23
rmt_suspendo( )		extern int	66–18
rmt_unlock( )		extern void	66–10
ru_ptr		extern volatile unsigned char *	73–3
		– S –	
s_conds		extern unsigned short	62–27
send_d_frame(count, struct_send_string_ptr, xmit_tag)		extern void	75–2 75–8
send_d_frame_il(il_buffer_number, relay_baton, data_start_offset, transmit_tag)		extern void	75–9
send_dl_prmtv_above(il_buffer_number, l2_relay_baton, l2_data_start_offset, size, l2_code, path, parameter)		extern void	62–54
send_dl_prmtv_below(il_buffer_number, l3_relay_baton, l3_data_start_offset, size, l3_code, path, parameter)		extern void	62–64
send_done_key_to_appl(flag)		extern void	68–19
send_frame(il_buffer_number, relay_baton, data_start_offset, transmit_frame_ptr)		extern void	70–12 72–16 73–5 76–12
send_frame_structure	(X.25 Layer 2)	struct	70–2
	(SNA)	struct	73–1
	(SDLC)	struct	72–2
addr_type		unsigned char	
frame_type		unsigned char	
NR_type		unsigned char	
NS_type		unsigned char	
p_f_type		unsigned char	
bcc_type		unsigned char	
addr_value		unsigned char	
cntrl_byte		unsigned char	
NR_value		unsigned char	
NS_value		unsigned char	

Identifier	Declaration	Page
send_frame_structure (LAPD)	struct	76–2
sapi_type	unsigned char	
tei_type	unsigned char	
cr_type	unsigned char	
frame_type	unsigned char	
NR_type	unsigned char	
NS_type	unsigned char	
p_f_type	unsigned char	
bcc_type	unsigned char	
sapi_value	unsigned char	
tei_value	unsigned char	
cntrl_byte	unsigned char	
NR_value	unsigned char	
NS_value	unsigned char	
send_key(number_of_keys, key_ptr)	extern void	68–18
send_m_dl_prmtv_above(il_buffer_number, l2_relay_baton, l2_data_start_offset,size, l2_code, path, parameter)	extern void	62–56
send_m_n_prmtv_above(il_buffer_number, l3_relay_baton, l3_data_start_offset, size, l3_code, path, parameter)	extern void	62–64
send_m_p_prmtv_above(il_buffer_number, l6_relay_baton, l6_data_start_offset, size, l6_code, path, parameter)	extern void	62–84
send_m_ph_prmtv_above(il_buffer_number, relay_baton, data_start_offset, size, code, path, parameter)	extern void	62–53
send_m_s_prmtv_above(il_buffer_number, l5_relay_baton, l5_data_start_offset, size, l5_code, path, parameter)	extern void	62–77
send_m_t_prmtv_above(il_buffer_number, l4_relay_baton, l4_data_start_offset, size, l4_code, path, parameter)	extern void	62–70
send_n_prmtv_above(il_buffer_number, l3_relay_baton, l3_data_start_offset, size, l3_code, path, parameter)	extern void	62–62
send_n_prmtv_below(il_buffer_number, l4_relay_baton, l4_data_start_offset, size, l4_code, path, parameter)	extern void	62–71
send_p_prmtv_above(il_buffer_number, l6_relay_baton, l6_data_start_offset, size, l6_code, path, parameter)	extern void	62–83
send_p_prmtv_below(il_buffer_number, relay_baton, data_start_offset, size, code, path, parameter)	extern void	62–89
send_packet(il_buffer_number, relay_baton, data_start_offset, transmit_packet_ptr)	extern void	71–16

Identifier	Declaration	Page
send_packet_structure	struct	71–2
path_num	unsigned char	
packet_type	unsigned char	
packet_type_byte	unsigned char	
m_bit	unsigned char	
d_bit	unsigned char	
q_bit	unsigned char	
pr_type	unsigned char	
ps_type	unsigned char	
pr_value	unsigned char	
ps_value	unsigned char	
cause	unsigned char	
diag_flag	unsigned char	
diag	unsigned char	
spare	unsigned char	
facilities_len	unsigned char	
facilities	char *	
data_len	unsigned short	
data	char *	
send_ph_prmtv_above(il_buffer_number, l2_relay_baton, l2_data_start_offset, size, l2_code, path, parameter)	extern void	62–51
send_ph_prmtv_below(il_buffer_number, l2_relay_baton, l2_data_start_offset, size, l2_code, path, parameter)	extern void	62–56
send_s_prmtv_above(il_buffer_number, l5_relay_baton, l5_data_start_offset, size, l5_code, path, parameter)	extern void	62–76
send_s_prmtv_below(il_buffer_number, l6_relay_baton, l6_data_start_offset, size, l6_code, path, parameter)	extern void	62–85
send_stat_message(struct_stat_msg_ptr)	extern void	61–15
send_t_prmtv_above(il_buffer_number, l4_relay_baton, l4_data_start_offset, size, l4_code, path, parameter)	extern void	62–69
send_t_prmtv_below(il_buffer_number, l5_relay_baton, l5_data_start_offset, size, l5_code, path, parameter)	extern void	62–78
set_a_conds(conditions)	extern unsigned short	62–92
set_aux_ctl_leads(ctl_or_not)	extern void	67–6
set_aux_direction(input_or_output)	extern void	67–5
set_aux_reg(reg_value_word)	extern void	67–10
set_cpm_date_time(year, month, day, hour, minute, second)	extern int	68–22
set_dl_conds(conditions)	extern unsigned short	62–59
set_dw_fkey_label(fkey, label_ptr)	extern void	60–24
_set_file_type(pathname_ptr, type_buff_ptr)	extern int	64–34
_set_hex_led(status)	extern void	60–30
set_isdn_speaker_chan(selection)	extern void	81–14
set_ltrace_fkey_label(layer, label_ptr)	extern void	60–51
set_m_a_conds(conditions)	extern unsigned short	62–92
set_m_dl_conds(conditions)	extern unsigned short	62–61
set_m_n_conds(conditions)	extern unsigned short	62–68

Identifier	Declaration	Page
set_m_p_conds(conditions)	extern unsigned short	62–89
set_m_s_conds(conditions)	extern unsigned short	62–82
set_m_t_conds(conditions)	extern unsigned short	62–75
_set_maint_buff_bit(il_buffer_number, new_bit_ptr)	extern void	62–44
set_max_frame_length(max_frame_length)	unsigned short	80–5
set_n_conds(conditions)	extern unsigned short	62–67
set_p_conds(conditions)	extern unsigned short	62–88
set_print_header(format_ptr)	extern int	63–8
set_s_conds(conditions)	extern unsigned short	62–81
set_t_conds(conditions)	extern unsigned short	62–74
set_tcr_b(tcr_register_mask, tcr_register_value)	extern void	58–16 69–9
set_tim_setup	extern void	81–2
set_utrace_fkey_label(buffer, label_ptr)	extern void	60–40
show_dw_fkey_labels( )	extern void	60–26
signal(signal_name)	extern void	68–15
signal_name	extern event	68–4
slip_bit( )	unsigned int	81–16
slip_dce_bit( )	unsigned int	81–17
slip_dte_bit( )	unsigned int	81–16
smds_format	extern unsigned char	82–3
smds_il_buffer	struct	82–1
lock	unsigned short	
maintain_bits	unsigned short	
buffer_size	unsigned short	
smds_resync_count	unsigned short	
receive_tag	unsigned short	
cell_busy_mask	unsigned short	
cell_crc10_error_mask	unsigned short	
reserved	unsigned long	
tick_count_high	unsigned short	
tick_count_mid	unsigned short	
tick_count_low	unsigned short	
available_space_offset	unsigned short	
bytes_remaining	unsigned short	
reserved	unsigned long	
data [1]	unsigned char	
sound_alarm( )	extern void	68–16
sprintf(string_ptr, format_ptr)	extern int	63–7
ss7_compression_mode	extern unsigned char	78–9
_start_il_buff_list(il_buffer_number, start_offset_ptr)	extern void	62–38
start_of_run_date	extern unsigned long	61–10
start_of_run_time	extern unsigned long	61–10
start_rcrd_play( )	extern void	68–16

Identifier	Declaration	Page
stat_msg	struct	61–5
op_type	unsigned short	
type	unsigned short	
object_name	unsigned long	
object_address	unsigned long	
stracef(array_ptr, string_ptr)	extern void	60–37
strcmp (user_string_ptr, line_data_ptr)	int	57–6
surrender_cpu( )	extern void	68–19
suspend_rcrd_play( )	extern void	68–17
	– T –	
t_conds	extern unsigned short	62–23
td_l_resyncs	extern unsigned short	83–4
td_l_smds_out_frame	extern unsigned char	82–3 83–4
td_buffer_off	extern unsigned short	58–5 74–2
td_il_buffer	extern unsigned short	58–5 74–2
td_modifier	extern unsigned char	58–5
temporary_prompt (string_pointer, seconds)	void	57–4
th_ptr	extern volatile unsigned char *	73–3
timeout	struct	68–3
event_id	unsigned long	
event_id_uid	unsigned short	
*timeout*_name_*expired*	extern event	68–4
timeout_restart_action(timeout_name_ptr, value, function)	extern void	68–8
timeout_stop_action(timeout_name_ptr)	extern void	68–10
timer_struct	struct	61–9
current	unsigned long	
last	unsigned long	
maximum	unsigned long	
minimum	unsigned long	
sample_count	unsigned short	
total_high	unsigned long	
total_low_low	unsigned short	
total_low_high	unsigned short	
start_tick_value	unsigned long	
running	unsigned short	
changed	unsigned short	
timestamp_struct	struct	68–3
timestamp_word_low	unsigned short	
timestamp_word_middle	unsigned short	
timestamp_word_high	unsigned short	

Identifier	Declaration	Page
tm	extern struct	68–2
tm_sec	int	
tm_min	int	
tm_hour	int	
tm_mday	int	
tm_mon	int	
tm_year	int	
tm_wday	int	
tm_yday	int	
tm_isdst	int	
trace_buf	extern struct	60–32
hdr	struct trace_buffer_header	
array [4096]	unsigned long	
trace_buffer_header	extern struct	60–32
logical_end	unsigned short	
logical_end_wrap_count	unsigned short	
modifier	unsigned char	
color	unsigned char	
enhance	unsigned char	
write_lock	unsigned short	
array_size	unsigned short	
line_size	unsigned char	
spare	unsigned char	
tracec(trace_buffer_ptr, character)	extern void	60–35
tracef(trace_buffer_ptr, format_ptr, . . .)	extern int	60–36
traces(trace_buffer_ptr, string_ptr)	extern void	60–39
	– U –	
ungetc(character, stream_ptr)	extern int	64–21
unhighlight_dw_fkey_label(fkey)	extern void	60–27

Identifier	Declaration	Page
unit_config	extern struct	65–3
disk_exists_mask	unsigned char	
hard_disk	unsigned char	
test_board	unsigned char	
mux	unsigned char	
modem	unsigned char	
num_mpms	unsigned char	
mpm [4]	struct mpm_info	
cpm_rev	unsigned char	
gbm_rev	unsigned char	
pcm_rev	unsigned char	
modem_rev	unsigned char	
mux_rev	unsigned char	
type_of_tim	unsigned char	
last_ram_cpm	unsigned long	
self_test_errors	unsigned long	
version	unsigned long	
model_number	unsigned long	
feb_type	unsigned char	
is_turbo	unsigned char	
xdram_rev_num	unsigned char	
xdram_present	unsigned char	
xdram_lo_addr	unsigned long	
xdram_hi_addr	unsigned long	
xdram_st_not_run	unsigned char	
hard_disk_type	unsigned char	
xtim_installed	unsigned char	
xsys_ram_present	unsigned char	
xsys_ram_lo_addr	unsigned long	
xsys_ram_hi_addr	unsigned long	
is_remote	unsigned char	
tim_mux_rev	unsigned short	
tim_mux_type	unsigned short	
is_upgrade	unsigned char	
spare1	unsigned char	
spare2	unsigned long	
spare3	unsigned long	
spare4	unsigned long	
spare5	unsigned long	
spare6	unsigned long	
sw_version	unsigned long	
fw_version	unsigned long	

Identifier	Declaration	Page
unit_setup	extern struct	65–2
speed_dce	unsigned long	
speed_dte	unsigned long	
usec_per_tick	unsigned long	
bit_order_parity	unsigned char	
bits_per_byte	unsigned char	
clocking_type	unsigned char	
data_source	unsigned char	
format	unsigned char	
mode	unsigned char	
parity	unsigned char	
code_name [13]	unsigned char	
record_capture_memory_type	unsigned char	
record_disk_drive	unsigned char	
record_data_type	unsigned char	
record_ram_speed	unsigned char	
record_initial _condition	unsigned char	
record_stop_at	unsigned char	
ticks_per_second	unsigned long	
unlock(lock_variable_ptr)	extern void	64–16 68–15
up_dl_il_buff	extern volatile unsigned short	62–18
up_dl_pdu_seg	extern volatile unsigned short	62–17
up_dl_prmtv	extern event	62–15
up_dl_prmtv_avail( )	extern unsigned short	62–59
up_dl_prmtv_code	extern volatile const unsigned char	62–18
up_dl_prmtv_path	extern volatile const unsigned char	62–18
up_dl_sdu	extern volatile unsigned short	62–18
up_n_il_buff	extern volatile unsigned short	62–22
up_n_pdu_seg	extern volatile unsigned short	62–21
up_n_prmtv	extern event	62–19
up_n_prmtv_avail( )	extern unsigned short	62–67
up_n_prmtv_code	extern volatile const unsigned char	62–22
up_n_prmtv_path	extern volatile const unsigned char	62–22
up_n_sdu	extern volatile unsigned short	62–22
up_p_il_buff	extern volatile unsigned short	62–34
up_p_pdu_seg	extern volatile unsigned short	62–33
up_p_prmtv	extern event	62–31
up_p_prmtv_avail( )	extern unsigned short	62–88
up_p_prmtv_code	extern volatile const unsigned char	62–33
up_p_prmtv_path	extern volatile const unsigned char	62–34
up_p_sdu	extern volatile unsigned short	62–34
up_ph_prmtv_avail( )	extern unsigned short	62–53
up_s_il_buff	extern volatile unsigned short	62–30
up_s_pdu_seg	extern volatile unsigned short	62–29
up_s_prmtv	extern event	62–27
up_s_prmtv_avail	extern unsigned short	62–81
up_s_prmtv_code	extern volatile const unsigned char	62–29

Identifier	Declaration	Page
up_s_prmtv_path	extern volatile const unsigned char	62–30
up_s_sdu	extern volatile unsigned short	62–30
up_t_il_buff	extern volatile unsigned short	62–26
up_t_pdu_seg	extern volatile unsigned short	62–25
up_t_prmtv	extern event	62–23
up_t_prmtv_avail	extern unsigned short	62–74
up_t_prmtv_code	extern volatile const unsigned char	62–25
up_t_prmtv_path	extern volatile const unsigned char	62–26
up_t_sdu	extern volatile unsigned short	62–26
user_setup_struct	struct	81–1
version	unsigned char	
tim_mux_rev	unsigned char	
data_channel	enum data_channel_type	
voice_channel	enum voice_channel_type	
	– W –	
window_color	extern unsigned short	60–6
window_modifier	extern unsigned char	60–8
write_aux(output_word)	extern void	67–7
	– X –	
x21_idle_action(character)	extern void	69–6
x21_transmit_call(count, struct_send_string_ptr, xmit_tag)	extern void	69–7
x21_transmit_call_idle(count, struct_send_string_ptr, xmit_tag, new_idle)	extern void	69–8
xmit_list	struct [Transmit]	58–1
	struct [X.21]	69–1
	struct [ISDN, Dual Port, Secondary Channel]	75–2
string	unsigned char *	
string_length	unsigned short	
xtim_load_program(slot,program_name)	extern void	81–19

Quick Reference Guide to the C Library

Identifier	Declaration	Page

C Library: Listing by Function

Accumulator

structures
- accumulator_struct

Alarm

routines
- sound_alarm

ATM

routine
- frame_disable_dce_1
- frame_disable_dce_2
- frame_disable_dte_1
- frame_disable_dte_2
- init_for_ds1
- init_for_e1

structures
- atm_il_buffer

variables
- atm_cells_per_bundle
- atm_framing_mode
- atm_mask_1
- atm_mask_2
- atm_mask_3
- atm_path_1
- atm_path_2
- atm_path_3
- atm_scrambling_selected
- hec_coset_value
- rd_1_resyncs
- rd_1_smds_out_frame
- td_1_resyncs
- td_1_smds_out_frame

Aux Port I/O

events
- aux_change

routines
- set_aux_ctl_leads
- set_aux_direction
- set_aux_reg
- write_aux

variables
- curr_aux_value
- prev_aux_value

Counter

events
- counter_name_change

structures
- counter_struct

Data Display

routines
- add_array_to_buff
- add_event_to_buff
- ctl_capture_rd
- ctl_capture_td
- ctl_enhance_rd
- ctl_enhance_td
- disable_dce
- disable_dte

variables
- rd_buffer_off
- rd_il_buffer
- rd_modifier
- td_buffer_off
- td_il_buffer
- td_modifier

DDCMP

events
- fevar_bd_bcc_rd
- fevar_bd_bcc_td
- fevar_bd_bcc2_rd
- fevar_bd_bcc2_td
- fevar_gd_bcc_rd
- fevar_gd_bcc_td
- fevar_gd_bcc2_rd
- fevar_gd_bcc2_td

variables
- rd_buffer_off
- rd_il_buffer
- td_buffer_off
- td_il_buffer

Disk

routines
- _raw_disk_read
- _raw_disk_write

Disk I/O

routines
- clearerr
- fclose
- feof
- ferror
- fflush
- fgetc
- fgets
- fopen
- fprintf
- fputc
- fputs
- fread
- fseek
- fwrite
- _get_file_type
- lock
- mkdir
- remove
- rename
- rewind
- _set_file_type
- ungetc
- unlock

Display Window

routines
- disable_freeze_disp_window
- display_prompt
- displayc
- displayf
- displays
- enable_freeze_disp_window
- highlight_dw_fkey_label
- pos_cursor
- restore_cursor
- set_dw_fkey_label
- _set_hex_led
- show_dw_fkey_labels
- sprintf
- unhighlight_dw_fkey_label

structures
- display_window_index_buffer

variables
- current_col
- current_line
- display_window_buffer
- window_color
- window_modifier

Dual−port

events
- d_dce_frame
- d_dte_frame
- d_rcv_frame,
- evar_abort_d_rd
- evar_abort_d_td
- evar_bd_bcc_d_rd
- evar_bd_bcc_d_td
- evar_flag_d_rd
- evar_flag_d_td
- evar_gd_bcc_d_rd
- evar_gd_bcc_d_td
- evar_rcvd_char_d_rd
- evar_rcvd_char_d_td
- fevar_abort_d_rd
- fevar_abort_d_td
- fevar_bd_bcc_d_rd
- fevar_bd_bcc_d_td
- fevar_d_dce_frame
- fevar_d_dte_frame
- fevar_d_rcv_frame
- fevar_d_xmit_cmplt
- fevar_flag_d_rd
- fevar_flag_d_td
- fevar_gd_bcc_d_rd
- fevar_gd_bcc_d_td
- fevar_rcvd_char_d_rd
- fevar_rcvd_char_d_td
- rcvd_char_d_rd
- rcvd_char_d_td

routines
- get_tim_setup
- send_d_frame
- send_d_frame_il
- set_tim_setup

structures
- xmit_list

variables
- d_channel_disabled

EIA

events
- fevar_eia_changed

routines
- ctl_eia

variables
- current_eia_leads
- previous_eia_leads

Flag

events
- flag_name_change

structures
- flag_struct

High Speed Frame Mode

events
- _chain_xmit
- compute_actual_length
- discard_d_dce_frames
- discard_d_dte_frames
- discard_dce_frames
- discard_dte_frames
- _high_speed_xmit

routines
- frame_disable_dce_1
- frame_disable_dce_2
- frame_disable_dte_1
- frame_disable_dte_2
- set_max_frame_length

Interrupt

events
- signal_name

routines
- signal

ISDN

events

- d_dce_frame
- d_dte_frame
- d_rcv_frame
- evar_abort_d_rd
- evar_abort_d_td
- evar_bd_bcc_d_rd
- evar_bd_bcc_d_td
- evar_flag_d_rd
- evar_flag_d_td
- evar_gd_bcc_d_rd
- evar_gd_bcc_d_td
- evar_rcvd_char_d_rd
- evar_rcvd_char_d_td
- fevar_abort_d_rd
- fevar_abort_d_td
- fevar_bd_bcc_d_rd
- fevar_bd_bcc_d_td
- fevar_d_dce_frame
- fevar_d_dte_frame
- fevar_d_rcv_frame
- fevar_d_xmit_cmplt
- fevar_flag_d_rd
- fevar_flag_d_td
- fevar_gd_bcc_d_rd
- fevar_gd_bcc_d_td
- fevar_rcvd_char_d_rd
- fevar_rcvd_char_d_td
- rcvd_char_d_rd
- rcvd_char_d_td

routines

- activate_link
- deactivate_link
- get_eoc_data *(S/T/U only)*
- get_tim_setup
- send_d_frame
- send_d_frame_il
- set_isdn_speaker_chan
- set_tim_setup

structures

- eoc_message *(S/T/U only)*
- user_setup_struct *(S/T/U only)*
- xmit_list

Keyboard

events

- keyboard_new_any_key
- keyboard_new_key

routines

- send_done_key_to_appl
- send_key

structures

- keyboard

variable

- keyboard_any_key

LAPD

events

- bcc_error
- dce_abort
- dce_bad_bcc
- dce_frame
- dce_good_bcc
- dte_abort
- dte_bad_bcc
- dte_frame
- dte_good_bcc
- frame_sent
- invalid_frame
- l2_T1
- nr_error
- ns_error
- rcvd_frame

routines

- l2_give_data
- resend_frame
- reset_nr
- reset_ns
- send_frame

structures

- send_frame_structure

variables

- l2_current_window_edge
- l2_enhance
- l2_lower_window_edge
- l2_resend_edge
- l2_suppress
- l2_upper_window_edge
- m_frame_addr_cr
- m_frame_addr_sapi
- m_frame_addr_tei
- m_frame_bcc_type
- m_frame_cntrl_byte_1
- m_frame_nr
- m_frame_ns
- m_frame_pf
- m_frame_type
- rcvd_frame_addr_cr
- rcvd_frame_addr_sapi
- rcvd_frame_addr_tei
- rcvd_frame_bcc_type
- rcvd_frame_buff_seg
- rcvd_frame_cntrl_byte_1
- rcvd_frame_nr
- rcvd_frame_ns
- rcvd_frame_pf
- rcvd_frame_sdu_offset
- rcvd_frame_sdu_size
- rcvd_frame_type

Line

events

fevar_abort_rd
fevar_abort_td
fevar_bd_bcc_rd
fevar_bd_bcc_td
fevar_flag_rd
fevar_flag_td
fevar_frm_error_rd
fevar_frm_error_td
fevar_gd_bcc_rd
fevar_gd_bcc_td
fevar_parity_rd
fevar_parity_td
fevar_rcv_buffer_full
fevar_rcvd_char_rd
fevar_rcvd_char_td

routines

ctl_capture_d_rd
ctl_capture_d_td
outsync_action

variables

rcv_buffer_full
rcvd_char_rd
rcvd_char_td

OSI

events

fevar_lo_dl_prmtv
fevar_lo_n_prmtv
fevar_lo_p_prmtv
fevar_lo_ph_prmtv
fevar_lo_s_prmtv
fevar_lo_t_prmtv
fevar_m_lo_dl_prmtv
fevar_m_lo_n_prmtv
fevar_m_lo_p_prmtv
fevar_m_lo_ph_prmtv
fevar_m_lo_s_prmtv
fevar_m_lo_t_prmtv
fevar_up_dl_prmtv
fevar_up_n_prmtv
fevar_up_p_prmtv
fevar_up_s_prmtv
fevar_up_t_prmtv
lo_dl_prmtv
lo_n_prmtv
lo_p_prmtv
lo_ph_prmtv
lo_s_prmtv
lo_t_prmtv
m_lo_dl_prmtv
m_lo_n_prmtv
m_lo_p_prmtv
m_lo_ph_prmtv
m_lo_s_prmtv
m_lo_t_prmtv
up_dl_prmtv
up_n_prmtv
up_p_prmtv
up_s_prmtv
up_t_prmtv

routines

_append_il_buff_list_cnt
_dup_il_buff_list
_dup_il_buff_list_start
_free_il_msg_buff
_get_il_msg_buff
_insert_il_buff_list_cnt
lo_dl_prmtv_avail
lo_n_prmtv_avail
lo_p_prmtv_avail
lo_ph_prmtv_avail
lo_s_prmtv_avail
lo_t_prmtv_avail
m_lo_dl_prmtv_avail
m_lo_n_prmtv_avail
m_lo_p_prmtv_avail
m_lo_ph_prmtv_avail
m_lo_s_prmtv_avail
m_lo_t_prmtv_avail
_open_space_in_il_buff
send_dl_prmtv_above
send_dl_prmtv_below
send_m_dl_prmtv_above
send_m_n_prmtv_above
send_m_p_prmtv_above
send_m_ph_prmtv_above
send_m_s_prmtv_above
send_m_t_prmtv_above
send_n_prmtv_above
send_n_prmtv_below
send_p_prmtv_above
send_p_prmtv_below
send_ph_prmtv_above
send_ph_prmtv_below
send_s_prmtv_above
send_s_prmtv_below
send_t_prmtv_above
send_t_prmtv_below
set_a_conds
set_dl_conds
set_m_a_conds
set_m_dl_conds
set_m_n_conds
set_m_p_conds
set_m_s_conds
set_m_t_conds
_set_maint_buff_bit
set_n_conds
set_p_conds
set_s_conds
set_t_conds
_start_il_buff_list
up_dl_prmtv_avail
up_n_prmtv_avail
up_p_prmtv_avail
up_ph_prmtv_avail
up_s_prmtv_avail
up_t_prmtv_avail

OSI (continued)

structures

il_buffer
il_list_header
il_list_node
pdu

variables

a_conds
dl_conds
l2_tick_count
l3_tick_count
l4_tick_count
l5_tick_count
l6_tick_count
l7_tick_count
lo_dl_il_buff
lo_dl_pdu_seg
lo_dl_prmtv_code
lo_dl_prmtv_path
lo_dl_sdu
lo_n_il_buff
lo_n_pdu_seg
lo_n_prmtv_code
lo_n_prmtv_path
lo_n_sdu
lo_p_il_buff
lo_p_pdu_seg
lo_p_prmtv_code
lo_p_prmtv_path
lo_p_sdu
lo_ph_il_buff
lo_ph_pdu_seg
lo_ph_prmtv_code
lo_ph_prmtv_path
lo_ph_sdu
lo_s_il_buff
lo_s_pdu_seg
lo_s_prmtv_code
lo_s_prmtv_path
lo_s_sdu
lo_t_il_buff
lo_t_pdu_seg
lo_t_prmtv_code
lo_t_prmtv_path
lo_t_sdu
m_a_conds
m_dl_conds
m_lo_dl_il_buff
m_lo_dl_pdu_seg
m_lo_dl_prmtv_code
m_lo_dl_prmtv_path
m_lo_dl_sdu_offset
m_lo_dl_sdu_size
m_lo_n_il_buff
m_lo_n_pdu_seg
m_lo_n_prmtv_code
m_lo_n_prmtv_path
m_lo_n_sdu_offset
m_lo_n_sdu_size
m_lo_p_il_buff
m_lo_p_pdu_seg
m_lo_p_prmtv_code
m_lo_p_prmtv_path
m_lo_p_sdu_offset
m_lo_p_sdu_size
m_lo_ph_il_buff
m_lo_ph_pdu_seg
m_lo_ph_prmtv_code
m_lo_ph_prmtv_path
m_lo_ph_sdu_offset
m_lo_ph_sdu_size
m_lo_s_il_buff
m_lo_s_pdu_seg
m_lo_s_prmtv_code
m_lo_s_prmtv_path
m_lo_s_sdu_offset
m_lo_s_sdu_size
m_lo_t_il_buff
m_lo_t_pdu_seg
m_lo_t_prmtv_code
m_lo_t_prmtv_path
m_lo_t_sdu_offset
m_lo_t_sdu_size
m_n_conds
m_p_conds
m_s_conds
m_t_conds
n_conds
p_conds
ph_ch2_discard_ind
ph_discard_ind
ph_prmtv_type
s_conds
t_conds
up_dl_il_buff
up_dl_pdu_seg
up_dl_prmtv_code
up_dl_prmtv_path
up_dl_sdu
up_n_il_buff
up_n_pdu_seg
up_n_prmtv_code
up_n_prmtv_path
up_n_sdu
up_p_il_buff
up_p_pdu_seg
up_p_prmtv_code
up_p_prmtv_path
up_p_sdu
up_s_il_buff
up_s_pdu_seg
up_s_prmtv_code
up_s_prmtv_path
up_s_sdu
up_t_il_buff
up_t_pdu_seg
up_t_prmtv_code
up_t_prmtv_path
up_t_sdu

Playback

routines
- start_rcrd_play
- suspend_rcrd_play

#pragma
- #pragma frame_nodes n
- #pragma hook
- #pragma il_buffers
- #pragma il_buffer_size
- #pragma layer
- #pragma nowarn
- #pragma object
- #pragma quad_display
- #pragma sample_mode
- #pragma set_chain_xmit
- #pragma set_high_speed_xmit
- #pragma smds_format E1
- #pragma smds_format DS1
- #pragma suppress_char_data
- #pragma traceburf

Print

routines
- printc
- printf
- prints
- reset_print_page
- set_print_header
- sprintf

structures
- print_buffer
- _print_buffer

Processing

routines
- bcopy
- lock
- surrender_cpu
- unlock

Program Chaining

routines
- load_program

Protocol Trace

routines
- set_ltrace_fkey_label

structures
- l2pp_trbuff_ctl
- l3pp_trbuff_ctl
- lpp_trbuff_ctl

variables
- l2pp_trbuff
- l2pp_trbuff_end
- l3pp_trbuff
- l3pp_trbuff_end

Q.931

events
- dce_packet
- dte_packet

variables
- l3_enhance
- l3_suppress
- m_call_ref_flag
- m_call_ref_len
- m_info_element_len
- m_message_type
- m_message_type_defined
- m_packet_bcc_type
- m_prot_disc
- m_ptr_to_call_ref
- m_ptr_to_info_element

Real–time Clock

events
- fevar_time_of_day

routines
- set_cpm_date_time

structures
- crnt_tm
- prev_tm
- tm

variables
- crnt_date_of_day
- crnt_time_of_day
- prev_date_of_day
- prev_time_of_day

Record

routines
- start_rcrd_play
- suspend_rcrd_play

Remote Port I/O

events

rmt_break
rmt_input_almost_empty
rmt_input_almost_full
rmt_input_empty
rmt_input_not_empty
rmt_input_overflow
rmt_output_empty

routines

rmt_flushi
rmt_flusho
rmt_get_baud_rate
rmt_get_bits
rmt_get_mode
rmt_get_parity
rmt_getc
rmt_getl
rmt_gets
rmt_lock
rmt_putb
rmt_putc
rmt_puts
rmt_resumeo
rmt_send_break
rmt_set_baud_rate
rmt_set_bits
rmt_set_mode
rmt_set_parity
rmt_suspendo
rmt_unlock

SDLC

events

bcc_error
dce_abort
dce_bad_bcc
dce_frame
dce_good_bcc
dte_abort
dte_bad_bcc
dte_frame
dte_good_bcc
frame_sent
invalid_frame
l2_T1
nr_error
ns_error
rcvd_frame

routines

l2_give_data
resend_frame
resend_frame_multi
reset_nr
reset_nr_multi
reset_ns
reset_ns_multi
send_frame

structures

l2_path_table[]
send_frame_structure

variables

l2_current_window_edge
l2_enhance
l2_lower_window_edge
l2_resend_edge
l2_suppress
l2_upper_window_edge
m_frame_addr
m_frame_bcc_type
m_frame_cntrl_byte_1
m_frame_pf
m_frame_type
rcvd_frame_addr
rcvd_frame_bcc_type
rcvd_frame_buff_seg
rcvd_frame_cntrl_byte_1
rcvd_frame_nr
rcvd_frame_ns
rcvd_frame_pf
rcvd_frame_sdu_offset
rcvd_frame_sdu_size
rcvd_frame_type

Secondary Channel

events

d_dce_frame
d_dte_frame
d_rcv_frame
evar_abort_d_rd
evar_abort_d_td
evar_bd_bcc_d_rd
evar_bd_bcc_d_td
evar_flag_d_rd
evar_flag_d_td
evar_gd_bcc_d_rd
evar_gd_bcc_d_td
evar_rcvd_char_d_rd
evar_rcvd_char_d_td
fevar_abort_d_rd
fevar_abort_d_td
fevar_bd_bcc_d_rd
fevar_bd_bcc_d_td
fevar_d_dce_frame
fevar_d_dte_frame
fevar_d_rcv_frame
fevar_d_xmit_cmplt
fevar_flag_d_rd
fevar_flag_d_td
fevar_gd_bcc_d_rd
fevar_gd_bcc_d_td
fevar_rcvd_char_d_rd
fevar_rcvd_char_d_td
rcvd_char_d_rd
rcvd_char_d_td

Secondary Channel (continued)

routines
- get_tim_setup
- send_d_frame
- send_d_frame_il
- set_tim_setup

structures
- xmit_list

variables
- d_channel_disabled

SMDS

routines
- frame_disable_dce_1
- frame_disable_dce_2
- frame_disable_dte_1
- frame_disable_dte_2
- init_for_ds1
- init_for_e1

structures
- smds_il_buffer

variables
- rd_1_smds_out_frame
- smds_format
- td_1_smds_out_frame

SNA

[Includes the entire SDLC library plus the following list of variables*.]*

variables
- m_packet_daf
- m_packet_def
- m_packet_dsaf
- m_packet_fi
- m_packet_fid_type
- m_packet_length
- m_packet_lsid
- m_packet_oaf
- m_packet_oef
- m_packet_osaf
- m_packet_rri
- m_packet_rti
- m_packet_ru_category
- m_packet_sdi
- rh_ptr
- ru_ptr
- th_ptr

SS#7 Layer 1

variables
- d_dce_frames_sup_run
- d_dce_frames_suppressed
- d_dte_frames_sup_run
- d_dte_frames_suppressed
- dce_flags
- dce_frames_sup_run
- dce_frames_suppressed
- dte_flags
- dte_frames_sup_run
- dte_frames_suppressed
- flags_d_rd
- flags_d_rd_max
- flags_d_rd_min
- flags_d_rd_run
- flags_d_td
- flags_d_td_max
- flags_d_td_min
- flags_d_td_run
- flags_rd
- flags_rd_max
- flags_rd_min
- flags_rd_run
- flags_td
- flags_td_max
- flags_td_min
- flags_td_run
- ss7_compression_mode

SS#7 Layer 2

events
- dce_abort
- dce_bad_bcc
- dce_frame
- dce_good_bcc
- dte_abort
- dte_bad_bcc
- dte_frame
- dte_good_bcc

variables
- l2_enhance
- l2_suppress
- m_bib
- m_fib
- m_frame_bcc_type
- m_li
- m_sn0
- m_unit_type

SS#7 Layer 3

events
- dce_packet
- dte_packet

variables
- l3_enhance
- l3_suppress
- m_cic
- m_code_type
- m_label_dpc
- m_label_opc
- m_label_sls
- m_sio_ni
- m_sio_priority
- m_sio_si

Stats Display

routines
- get_68k_phys_addr
- send_stat_message

structures
- stat_msg

Status

events
- display_screen_changed

structures
- mpm_info
- unit_config
- unit_setup

variables
- crnt_display_screen
- prev_display_screen

String

routines
- index
- rindex

Synchronization

routines
- slip_bit *
- slip_dce_bit *
- slip_dte_bit *

TIM Expansion Shelf

routines
- get_xtim_config
- xtim_load_program

Timeout

events
- timeout_name_expired

routines
- timeout_restart_action
- timeout_stop_action

structures
- timeout

Timer

routines
- convert_tick_count
- get_wall_time_286_ticks
- get_wall_time_ticks

structures
- timer_struct

variables
- l1_tick_count
- start_of_run_date
- start_of_run_time

Timestamp

routines
- get_user_timestamp

structures
- timestamp_struct

Trace Buffer

routines
- set_utrace_fkey_label
- stracef
- tracec
- tracef
- traces

structures
- l1_trbuf
- l2_trbuf
- l3_trbuf
- l4_trbuf
- l5_trbuf
- l6_trbuf
- l7_trbuf
- prog_trbuf
- trace_buf
- trace_buffer_header

Transmit

events
- fevar_d_xmit_cmplt
- fevar_xmit_cmplt

routines
- idle_action
- l1_il_transmit
- l1_transmit
- set_tcr_b

structures
- xmit_list

X.21

events
- fevar_eia_changed

routines
- ctl_eia
- enter_call_phase
- enter_data_phase
- set_tcr_b
- x21_idle_action
- x21_transmit_call
- x21_transmit_call_idle

structures
- xmit_list

variables
- current_eia_leads
- previous_eia_leads

* *slip_bit, slip_dce_bit, and slip_dte_bit synchronization routines are for use with Dual-Port DS-0A TIMs only.*

X.25 Layer 2

events

- bcc_error
- dce_abort
- dce_bad_bcc
- dce_frame
- dce_good_bcc
- dte_abort
- dte_bad_bcc
- dte_frame
- dte_good_bcc
- frame_sent
- invalid_frame
- l2_T1
- nr_error
- ns_error
- rcvd_frame

routines

- l2_give_data
- resend_frame
- reset_nr
- reset_ns
- send_frame

structures

- send_frame_structure

variables

- l2_current_window_edge
- l2_enhance
- l2_lower_window_edge
- l2_resend_edge
- l2_suppress
- l2_upper_window_edge
- m_frame_addr
- m_frame_bcc_type
- m_frame_cntrl_byte_1
- m_frame_pf
- m_frame_type
- rcvd_frame_addr
- rcvd_frame_bcc_type
- rcvd_frame_buff_seg
- rcvd_frame_cntrl_byte_1
- rcvd_frame_nr
- rcvd_frame_ns
- rcvd_frame_pf
- rcvd_frame_sdu_offset
- rcvd_frame_sdu_size
- rcvd_frame_type

X.25 Layer 3

events

- dce_packet
- dte_packet
- invalid_packet
- packet_sent
- pr_error
- ps_error
- rcvd_packet

routines

- l3_clear_path
- l3_give_data
- l3_more_to_resend
- l3_window_empty
- l3_window_full
- resend_packet
- reset_pr_ps
- send_packet

structure

- send_packet_structure

variables

- l3_enhance
- l3_suppress
- m_packet_buff_seg
- m_packet_cause
- m_packet_d
- m_packet_diag_code
- m_packet_info_length
- m_packet_info_offset
- m_packet_info_ptr
- m_packet_info_seg
- m_packet_lcn
- m_packet_lcn_grp
- m_packet_length
- m_packet_m
- m_packet_pr
- m_packet_ps
- m_packet_ptr
- m_packet_q
- m_packet_sdu_offset
- m_packet_type
- m_packet_type_byte
- rcvd_device_path
- rcvd_packet_ptr
- rcvd_packet_type
- rcvd_pkt_buff_seg
- rcvd_pkt_cause
- rcvd_pkt_d
- rcvd_pkt_diagn
- rcvd_pkt_info_length
- rcvd_pkt_info_offset
- rcvd_pkt_info_ptr
- rcvd_pkt_info_seg
- rcvd_pkt_lcn
- rcvd_pkt_length
- rcvd_pkt_m
- rcvd_pkt_pr
- rcvd_pkt_ps
- rcvd_pkt_q
- rcvd_pkt_sdu_offset
- rcvd_pkt_type_byte

C Library: Alphabetic Definitions

a_conds

Indicates whether or not Layer 7 will respond to OSI event conditions from Layer 6 (emulate mode only).

accumulator_struct

Structure of an accumulator.

activate_link

Initiates the activation process when the INTERVIEW is a TE device in the short passive bus configuration on the S/T interface; use with ISDN S/T/U interface only.

add_array_to_buff

Writes specified number of event words from array to 64–Kbyte character buffer.

add_event_to_buff

Writes 16–bit event word to 64–Kbyte character buffer.

_append_il_buff_list_cnt

Appends text node at the end of a linked list of text inside IL buffer.

atm_cells_per_bundle

Contains the number of ATM cells in each bundle, depending upon the size of the IL buffer: 10, 12, 14, or 16; READ ONLY variable valid when HEC framing is selected.

atm_framing_mode

Indicates the framing mode, HEC cell delineation or PLCP framing; READ ONLY variable.

atm_il_buffer

Structure of an ATM interlayer–message buffer, both directions.

atm_mask_1

Determines which bits of the user variable *atm_path_1* are to be compared with each cell header.

atm_mask_2

Determines which bits of the user variable *atm_path_2* are to be compared with each cell header.

atm_mask_3

Determines which bits of the user variable *atm_path_3* are to be compared with each cell header.

atm_path_1

Header value that is to be tracked by the Layer 1 code. A bit-mask word is allocated in the IL buffer header *atm_path_mask1* for this path. The bit corresponding to the cell's position within the cell bundle will be 1 if the cell's header matches that in this variable, 0 otherwise.

atm_path_2

Header value that is to be tracked by the Layer 1 code. A bit-mask word is allocated in the IL buffer header *atm_path_mask2* for this path.

atm_path_3

Header value that is to be tracked by the Layer 1 code. A bit-mask word is allocated in the IL buffer header *atm_path_mask3* for this path.

atm_scrambling_selected

Selects scrambling; READ ONLY variable.

aux_change

Event true when status of AUX port lead designated as control and input changes.

bcc_error

Event true when BCC is detected (emulate mode only).

bcopy

Executes a very fast byte-by-byte copy of a specified number of bytes from a source variable to a destination variable.

_chain_xmit

Indicates to the application program whether or not the Chain Transmit mode is on for each transmit channel: the first subscript is for Channel 1 and the second subscript is for Channel 2 where *0* is OFF and *1* is ON.

clearerr

Clears the end-of-file and error indicators for a stream.

compute_actual_length

Provides the total number of characters in each frame in the header of the il message buffer that contains the frame.

convert_tick_count

Converts a designated tick count into microseconds.

*counter*_name_*change*

Event true when named counter is incremented, decremented, or set to a new value.

counter_struct
Structure of a counter.

crnt_date_of_day
Stores current day of the month.

crnt_display_screen
Stores current display screen and indicates Freeze or Run mode.

crnt_time_of_day
Stores current time of day.

crnt_tm
Structure of current time of day. Instance of structure *tm*.

ctl_capture_d_rd
In dual–channel or dual–port operation, turns on/off storing of DCE data for the secondary channel.

ctl_capture_d_td
In dual–channel or dual–port operation, turns on/off storing of DTE data for the secondary channel.

ctl_capture_rd
Turns on/off presentation of DCE data to the data–display screen and capture of data to the screen buffer.

ctl_capture_td
Turns on/off presentation of DTE data to the data–display screen and capture of data to the screen buffer.

ctl_eia
Controls status of up to 3 of 9 leads. In X.21, controls two control leads.

ctl_enhance_rd
Controls enhancements of data display on DCE side of line.

ctl_enhance_td
Controls enhancements of data display on DTE side of line.

curr_aux_value
Status of AUX port leads. Result of this value exclusive *or*ed (^) with *prev_aux_value* indicates those leads whose status has changed.

current_col

Stores current column number of cursor position in Display Window.

current_eia_leads

Stores status of EIA or X.21 leads (when enabled on FEB setup).

current_line

Stores current row number of cursor position in Display Window.

d_channel_disabled

Identifies whether or not both ports are being used when unit is configured for dual–port operation.

d_dce_frame

Event true when a DCE frame is detected on the secondary channel.

d_dce_frames_sup_run

For secondary channel in dual–port or dual–channel operation, the number of DCE Fill–In or Link Status Signal Units suppressed since *d_dce_frames_suppressed* was last updated (SS#7).

d_dce_frames_suppressed

For secondary channel in dual–port or dual–channel operation, the number of DCE Fill–In or Link Status Signal Units suppressed since the last frame (SS#7).

d_dte_frame

Event true when a DTE frame is detected on the secondary channel.

d_dte_frames_sup_run

For secondary channel in dual–port or dual–channel operation, the number of DTE Fill–In or Link Status Signal Units suppressed since *d_dte_frames_suppressed* was last updated (SS#7).

d_dte_frames_suppressed

For secondary channel in dual–port or dual–channel operation, the number of DTE Fill–In or Link Status Signal Units suppressed since the last frame (SS#7).

d_rcv_frame

Event true when a frame is received on the secondary channel (emulate mode only).

dce_abort

Event true when an abort is detected for a DCE frame.

dce_bad_bcc
Event true when a bad BCC is calculated for a DCE frame.

dce_flags
Previous name of *flags_rd* variable. See *flags_rd* (SS#7).

dce_frame
Event true when a DCE frame is detected.

dce_frames_sup_run
Stores the number of DCE Fill-In or Link Status Signal Units suppressed since *dce_frames_suppressed* was last updated (SS#7).

dce_frames_suppressed
Stores the number of DCE Fill-In or Link Status Signal Units suppressed since the last frame (SS#7).

dce_good_bcc
Event true when a good BCC is calculated for a DCE frame.

dce_packet
Event true when a DCE packet is detected.

deactivate_link
Initiates the deactivation process when the INTERVIEW is a TE device in the short passive bus configuration on the S/T interface; use with ISDN S/T/U interface only.

disable_dce
Disables monitoring of DCE side of line (SYNC format; emulate mode).

disable_dte
Disables monitoring of DTE side of line (SYNC format; emulate mode).

disable_freeze_disp_window
Disables operation of FREEZE key during Run mode for the Display Window only.

discard_d_dce_frames
Event true for frames from the associated channel which are then not passed to Layer 2; setting to False will then pass the frames to Layer 2.

discard_d_dte_frames
Event true for frames from the associated channel which are then not passed to Layer 2; setting to False will then pass the frames to Layer 2.

discard_dce_frames

Event true for frames from the associated channel which are then not passed to Layer 2; setting to False will then pass the frames to Layer 2.

discard_dte_frames

Event true for frames from the associated channel which are then not passed to Layer 2; setting to False will then pass the frames to Layer 2.

display_binary

Takes user-defined, 8-bit values as input and expands them into a binary display of ASCII 1's and 0's.

display_prompt

Displays a designated string at the beginning of the Display Window's prompt line.

display_screen_changed

Event true when Run-mode display screen is changed or when Run/Freeze mode is changed.

display_window_buffer

Array of 32-bit words (3 bytes attributes, 1-byte character) that comprise Display Window (1088 elements).

display_window_index_buffer

Array of structures used to detect changes to the Display Window buffer.

displayc

Outputs a single ASCII character to the Display Window.

displayf

Writes output, with or without attributes, to the Display Window according to a string that specifies how arguments should be converted for output.

displays

Writes output to the Display Window under control of a string.

dl_conds

Indicates whether or not Layer 2 will respond to OSI event conditions from Layers 1 and/or 3 (emulate mode only).

dte_abort

Event true when an abort is detected for a DTE frame.

dte_bad_bcc
Event true when a bad BCC is calculated for a DTE frame.

dte_flags
Previous name of *flags_td* variable. See *flags_td* (SS#7).

dte_frame
Event true when a DTE frame is detected.

dte_frames_sup_run
Stores the number of DTE Fill-In or Link Status Signal Units suppressed since *dte_frames_suppressed* was last updated (SS#7).

dte_frames_suppressed
Stores the number of DTE Fill-In or Link Status Signal Units suppressed since the last frame (SS#7).

dte_good_bcc
Event true when a good BCC is calculated for a DTE frame.

dte_packet
Event true when a DTE packet is detected.

_dup_il_buff_list
Duplicates entire pointer list so that lower layer can have its own list (for resends) and *append* data to the IL buffer.

_dup_il_buff_list_start
Duplicates header node of a pointer list so that lower layer can start its own list (for resends) to *insert* data in the IL buffer.

enable_freeze_disp_window
Enables normal operation of FREEZE key during Run mode for the Display Window only.

enter_call_phase
Overrides existing selections of Line Setup menu with ASCII code, 7-bit odd parity, and SYNC format during X.21 call-establishment phase.

enter_data_phase
Implements existing selections of Line Setup menu during X.21 data-transfer phase.

eoc_message
Structure of an ISDN S/T/U status message.

evar_abort_d_rd
Event true when an abort is detected in an RD frame on the secondary channel (emulate or monitor mode).

evar_abort_d_td
Event true when an abort is detected in a TD frame on the secondary channel (emulate or monitor mode).

evar_bd_bcc_d_rd
Event true when a bad BCC is calculated for an RD block or frame on the secondary channe (emulate or monitor mode).

evar_bd_bcc_d_td
Event true when a bad BCC is calculated for a TD block or frame on the secondary channel (emulate or monitor mode).

evar_flag_d_rd
Event true when a BOP flag is detected on the RD side of the line on the secondary channel (emulate or monitor mode).

evar_flag_d_td
Event true when a BOP flag is detected on the TD side of the line on the secondary channel (emulate or monitor mode).

evar_gd_bcc_d_rd
Event true when a good BCC is calculated for an RD block or frame on the secondary channel (emulate or monitor mode).

evar_gd_bcc_d_td
Event true when a good BCC is calculated for a TD block or frame on the secondary channel (emulate or monitor mode).

evar_rcvd_char_d_rd
Event true for each character received on RD on the secondary channel (emulate or monitor mode).

evar_rcvd_char_d_td
Event true for each character received on TD on the secondary channel (emulate or monitor mode).

fclose
Disassociates stream from open input/output file and closes the disk file.

feof
Tests the end-of-file indicator for a stream.

ferror
Test the error indicator for a stream.

fevar_abort_d_rd
Event true when an abort is detected in an RD frame on the secondary channel (emulate or monitor mode).

fevar_abort_d_td
Event true when an abort is detected in a TD frame on the secondary channel (emulate or monitor mode).

fevar_abort_rd
Event true when an abort is detected in an RD frame.

fevar_abort_td
Event true when an abort is detected in a TD frame.

fevar_bd_bcc_d_rd
Event true when a bad BCC is calculated for an RD block or frame on the secondary channel (emulate or monitor mode).

fevar_bd_bcc_d_td
Event true when a bad BCC is calculated for a TD block or frame on the secondary channel (emulate or monitor mode).

fevar_bd_bcc_rd
Event true when a bad BCC is calculated for an RD block or frame. In DDCMP, true when a bad *header* BCC received on RD.

fevar_bd_bcc_td
Event true when a bad BCC is calculated for a TD block or frame. In DDCMP, true when a bad *header* BCC received on TD.

fevar_bd_bcc2_rd
Event true when a bad *data* BCC received on RD (DDCMP).

fevar_bd_bcc2_td
Event true when a bad *data* BCC received on TD (DDCMP).

fevar_d_dce_frame
Event true when a DCE frame is detected on the secondary channel.

fevar_d_dte_frame
Event true when a DTE frame is detected on the secondary channel.

fevar_d_rcv_frame
Event true when a frame is received on the secondary channel.

fevar_d_xmit_cmplt
Event true when INTERVIEW puts a transmission out onto the secondary channel link.

fevar_eia_changed
Event true when status of EIA (non−data) or X.21 control lead changes.

fevar_flag_d_rd
Event true when a BOP flag is detected on the RD side of the line on the secondary channel (emulate or monitor mode).

fevar_flag_d_td
Event true when a BOP flag is detected on the TD side of the line on the secondary channel (emulate or monitor mode).

fevar_flag_rd
Event true when BOP flag is detected on RD side of the line.

fevar_flag_td
Event true when BOP flag is detected on TD side of the line.

fevar_frm_error_rd
Event true when an ASYNC framing error is detected for an RD byte.

fevar_frm_error_td
Event true when an ASYNC framing error is detected for a TD byte.

fevar_gd_bcc_d_rd
Event true when a good BCC is calculated for an RD block or frame on the secondary channel (emulate or monitor mode).

fevar_gd_bcc_d_td
Event true when a good BCC is calculated for a TD block or frame on the secondary channel (emulate or monitor mode).

fevar_gd_bcc_rd
Event true when a good BCC is calculated for an RD block or frame. In DDCMP, true when a good *header* BCC received on RD.

fevar_gd_bcc_td
Event true when a good BCC is calculated for a TD block or frame. In DDCMP, true when a good *header* BCC received on TD.

fevar_gd_bcc2_rd
Event true when a good *data* BCC received on RD (DDCMP).

fevar_gd_bcc2_td
Event true when a good *data* BCC received on TD (DDCMP).

fevar_lo_dl_prmtv
Event true when an OSI primitive is received at Layer 3 from Layer 2.

fevar_lo_n_prmtv
Event true when an OSI primitive is received at Layer 4 from Layer 3

fevar_lo_p_prmtv
Event true when an OSI primitive is received at Layer 7 from Layer 6.

fevar_lo_ph_prmtv
Event true when an OSI primitive is received at Layer 2 from Layer 1.

fevar_lo_s_prmtv
Event true when an OSI primitive is received at Layer 6 from Layer 5

fevar_lo_t_prmtv
Event true when an OSI primitive is received at Layer 5 from Layer 4.

fevar_m_lo_dl_prmtv
Event true when an OSI primitive is received at Layer 3 from Layer 2.

fevar_m_lo_n_prmtv
Event true when an OSI primitive is received at Layer 4 from Layer 3.

fevar_m_lo_p_prmtv
Event true when an OSI primitive is received at Layer 7 from Layer 6.

fevar_m_lo_ph_prmtv
Event true when an OSI primitive is received at Layer 2 from Layer 1.

fevar_m_lo_s_prmtv
Event true when an OSI primitive is received at Layer 6 from Layer 5.

fevar_m_lo_t_prmtv
Event true when an OSI primitive is received at Layer 5 from Layer 4.

fevar_parity_rd
Event true when parity error detected for an RD byte.

fevar_parity_td
Event true when parity error detected for a TD byte.

fevar_rcv_buffer_full
Event true when character buffer full with data. Overwriting begins.

fevar_rcvd_char_d_rd
Event true for each character received on RD on the secondary channel (emulate or monitor mode).

fevar_rcvd_char_d_td
Event true for each character received on TD on the secondary channel (emulate or monitor mode).

fevar_rcvd_char_rd
Event true for each character received on RD side of line.

fevar_rcvd_char_td
Event true for each character received on TD side of line.

fevar_time_of_day
Event true once per minute.

fevar_up_dl_prmtv
Event true when an OSI primitive is received at Layer 2 from Layer 3.

fevar_up_n_prmtv
Event true when an OSI primitive is received at Layer 3 from Layer 4.

fevar_up_p_prmtv
Event true when an OSI primitive is received at Layer 6 from Layer 7.

fevar_up_s_prmtv
Event true when an OSI primitive is received at Layer 5 from Layer 6.

fevar_up_t_prmtv
Event true when an OSI primitive is received at Layer 4 from Layer 5.

fevar_xmit_cmplt
Event true when INTERVIEW puts transmission out onto the link.

fflush
Causes unwritten data to be written to output file, or undoes effect of preceding *ungetc* operation on input file.

fgetc
Gets next character (if present) from input stream.

fgets
Gets at most one less than specified number of 8-bit characters from input stream and puts them in an array.

*flag*_name_*change*
Event signaled by program itself as part of a FLAG increment, decrement, or set action.

flag_struct
Structure of flag.

flags_d_rd
For secondary channel in dual-port or dual-channel operation, the number of compressed RD flags received since the last frame (SS#7).

flags_d_rd_max
For secondary channel in dual-port or dual-channel operation, the maximum number of compressed RD flags between frames (SS#7).

flags_d_rd_min
For secondary channel in dual-port or dual-channel operation, the minimum number of compressed RD flags between frames (SS#7).

flags_d_rd_run
For secondary channel in dual-port or dual-channel operation, the number of compressed RD flags received since *flags_d_rd* was last updated (SS#7).

flags_d_td
For secondary channel in dual-port or dual-channel operation, the number of compressed TD flags received since the last frame (SS#7).

flags_d_td_max
For secondary channel in dual-port or dual-channel operation, the maximum number of compressed TD flags between frames (SS#7).

flags_d_td_min
For secondary channel in dual-port or dual-channel operation, the minimum number of compressed TD flags between frames (SS#7).

flags_d_td_run
For secondary channel in dual-port or dual-channel operation, the number of compressed TD flags received since *flags_d_td* was last updated (SS#7).

flags_rd
Stores the number of compressed RD flags received since the last frame (SS#7).

flags_rd_max
Stores the maximum number of compressed RD flags between frames (SS#7).

flags_rd_min
Stores the minimum number of compressed RD flags between frames (SS#7).

flags_rd_run
Stores the number of compressed RD flags received since *flags_rd* was last updated (SS#7).

flags_td
Stores the number of compressed TD flags received since the last frame (SS#7).

flags_td_max
Stores the maximum number of compressed TD flags between frames (SS#7).

flags_td_min
Stores the minimum number of compressed TD flags between frames (SS#7).

flags_td_run
Stores the number of compressed TD flags received since *flags_td* was last updated (SS#7).

fopen
Opens a file for reading or writing.

fprintf
Writes specified output to an output stream according to a string that specifies how arguments should be converted for output.

fputc
Writes a given character to an output stream.

fputs
Writes a string of characters from an array to an output stream.

frame_disable_dce_1
Disables receipt of data by the INTERVIEW on the DCE side of the primary port (or channel), freeing up the CPU power for other tasks.

frame_disable_dce_2
Disables receipt of data by the INTERVIEW on the DTE side of the primary port (or channel), freeing up the CPU power for other tasks.

frame_disable_dte_1
Disables receipt of data by the INTERVIEW on the DCE side of the secondary port (or channel), freeing up the CPU power for other tasks.

frame_disable_dte_2
Disables receipt of data by the INTERVIEW on the DTE side of the secondary port (or channel), freeing up the CPU power for other tasks.

frame_sent
Event true when frame passed down to Layer 1 (emulate mode only).

fread
Reads elements of specified size from an input stream into a user-defined buffer.

_free_il_msg_buff
Returns IL buffer to the pool of free buffers.

fseek
Manipulates file-position indicator, clears end-of-file indicator, and resets *ungetc* variable.

fwrite
Writes elements of specified size from user-defined array to an output stream.

get_68k_phys_addr
Converts address of specified variable in 80286 processors (MPMs) to 68010 (CPM) format.

get_eoc_data
Accesses maintenance channel messages from the ISDN U interface and information on the state changes in the S/T or U interface link. (ISDN S/T/U TIM)

_get_file_type
Determines the label type of a specified file.

_get_il_msg_buff
Gets a free IL buffer from pool of available buffers. Returns buffer number to caller for use in subsequent calls to other IL buffer services and provides a maintain bit for the freeing operation.

get_tim_setup
Dynamically changes the B channel on which emulation or monitoring is being done. Use with *set_tim_setup*.

get_user_timestamp
Returns the timestamp that would be put in an IL buffer if an end of frame were to be received at the time that the routine was called. This routine must be declared.

get_wall_time_286_ticks
Gets the number of wall–time ticks in MPM format since Run mode was entered.

get_wall_time_ticks
Gets the number of wall–time ticks in CPM format since Run mode was entered.

get_xtim_config
Allows an application to read the information on a Test Interface Module when the TIM Expansion Shelf is installed in the INTERVIEW.

hec_coset_value
Displays the value selected in the **HEC CRC–8 Coset** field of the Line Setup screen; READ ONLY variable.

_high_speed_xmit
Indicates to the application program whether or not the High Speed Transmit mode is on for each transmit channel: the first subscript is for Channel 1 and the second subscript is for Channel 2 where *0* is OFF and *1* is ON.

highlight_dw_fkey_label
Displays a specified user–defined softkey label in the Display Window in reverse video.

idle_action
Changes the idle–line condition applied to SYNC data by the INTERVIEW.

il_buffer
Structure of an IL message buffer.

il_list_header
Structure of the header node in an IL message buffer in primitive sent down from the layer above.

il_list_node
Structure of text nodes in an IL message buffer in primitive sent down from the layer above.

index
Searches for an instance of a character starting from the beginning of a specified list.

init_for_ds1
Changes the SMDS receive format to DS1; can be invoked at any time. (SMDS, ATM)

init_for_e1
Changes the SMDS receive format to E1; can be invoked at any time. (SMDS, ATM)

_insert_il_buff_list_cnt
Inserts a text node at the beginning of a linked list inside an IL buffer.

invalid_frame
Event true when an invalid frame is detected (emulate mode only).

invalid_packet
Event true when an invalid packet is detected (emulate mode only).

keyboard
Structure that contains the ASCII value of key just executed.

keyboard_any_key
Stores value of last key or key–combination executed.

keyboard_new_any_key
Event true when any key (except CAPS LOCK) is pressed.

keyboard_new_key
Event true when any ASCII key is pressed.

l1_il_transmit
Sends a designated IL message buffer out onto the line.

l1_tick_count
Stores the number of ticks since start of Run mode.

l1_transmit
Sends a specified string with a user–determined BCC.

l1_trbuf
Structure of one of seven user trace buffers (TRACE 1). Instance of structure *trace_buf*.

l2_current_window_edge
Compare with other window–edge variables to determine status of retransmit window (emulate mode only).

l2_enhance
Stores current enhancement attribute for Layer 2 Protocol Trace display.

l2_give_data
Used in passing IL buffer associated with received frame up to Layer 3 (emulate mode only).

l2_lower_window_edge
Compare with other window–edge variables to determine status of retransmit window (emulate mode only).

l2_path_table
Array of structures for monitoring the transmit windows of multiple addresses in SDLC multi–drop operation (emulate mode only).

l2_resend_edge
Compare with other window–edge variables to determine if there is more in the resend window (emulate mode only).

l2_suppress
Determines whether or not frames specified in conditions will be displayed in the Layer 2 Protocol Trace.

l2_T1
Event true when the T1 timer has expired (emulate mode only).

l2_tick_count
Stores 32–bit *l1_tick_count* that is in header of most recent IL buffer passed up to Layer 2. Original time when end of data was clocked into the buffer.

l2_trbuf
Structure of one of seven user trace buffers (TRACE 2). Instance of structure *trace_buf*.

l2_upper_window_edge
Compare with other window–edge variables to determine status of retransmit window (emulate mode only).

l2pp_trbuff
Stores the first data byte in the Layer 2 Protocol Trace buffer.

l2pp_trbuff_ctl
Variables in this structure monitor the logical location in the Layer 2 Protocol Trace buffer.

l2pp_trbuff_end
Stores the first byte of the 2–byte trailer of the Layer 2 Protocol Trace buffer.

l3_clear_path
Resets P(R)– and P(S)–related variables, clears transmit window, and resets LCN and address fields to void on a designated path (X.25, emulate mode only).

l3_enhance
Stores current enhancement attribute for Layer 3 Protocol Trace display.

l3_give_data
Used in passing IL buffer associated with received data packet up to Layer 4 (X.25, emulate mode only).

l3_more_to_resend
Determines whether or not there are any more X.25 packets in the transmit window to resend (emulate mode only).

l3_suppress
Determines whether or not packets specified in conditions will be displayed in the Layer 3 Protocol Trace.

l3_tick_count
Stores 32–bit *l1_tick_count* that is in header of most recent IL buffer passed up to Layer 3. Original time when end of data was clocked into the buffer.

l3_trbuf
Structure of one of seven user trace buffers (TRACE 3). Instance of structure *trace_buf*.

l3_window_empty
Determines whether the Layer 3 window for a specified path is empty or not empty (X.25, emulate mode only).

l3_window_full
Determines whether the Layer 3 window for a specified path is full or not full (X.25, emulate mode only).

l3pp_trbuff
Stores the first data byte in the Layer 3 Protocol Trace buffer.

l3pp_trbuff_ctl
Variables in this structure monitor the logical location in the Layer 3 Protocol Trace buffer.

l3pp_trbuff_end
Stores the first byte of the 2–byte trailer of the Layer 3 Protocol Trace buffer.

l4_tick_count
Stores 32–bit *l1_tick_count* that is in header of most recent IL buffer passed up to Layer 4. Original time when end of data was clocked into the buffer.

l4_trbuf
Structure of one of seven user trace buffers (TRACE 4). Instance of structure *trace_buf*.

l5_tick_count
Stores 32–bit *l1_tick_count* that is in header of most recent IL buffer passed up to Layer 5. Original time when end of data was clocked into the buffer.

l5_trbuf
Structure of one of seven user trace buffers (TRACE 5). Instance of structure *trace_buf*.

l6_tick_count
Stores 32–bit *l1_tick_count* that is in header of most recent IL buffer passed up to Layer 6. Original time when end of data was clocked into the buffer.

l6_trbuf
Structure of one of seven user trace buffers (TRACE 6). Instance of structure *trace_buf*.

l7_tick_count
Stores 32–bit *l1_tick_count* that is in header of most recent IL buffer passed up to Layer 7. Original time when end of data was clocked into the buffer.

l7_trbuf
Structure of one of seven user trace buffers (TRACE 7). Instance of structure *trace_buf*.

lo_dl_il_buff
Stores IL buffer number received at Layer 3 in a PDU from Layer 2 (emulate mode only).

lo_dl_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 3 from Layer 2 (emulate mode only).

lo_dl_prmtv
Event true when OSI primitive received at Layer 3 from Layer 2 (emulate mode only).

lo_dl_prmtv_avail
Indicates whether one or more OSI primitives from Layer 2 are queued and available to be received at Layer 3 (emulate mode only).

lo_dl_prmtv_code
Stores code for OSI primitive received at Layer 3 in a PDU from Layer 2 (emulate mode only).

lo_dl_prmtv_path
Stores path number received at Layer 3 in a PDU from Layer 2 (emulate mode only).

lo_dl_sdu
Stores offset to where the SDU begins in an OSI primitive received at Layer 3 from Layer 2 (emulate mode only).

lo_n_il_buff
Stores IL buffer number received at Layer 4 in a PDU from Layer 3 (emulate mode only).

lo_n_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 4 from Layer 3 (emulate mode only).

lo_n_prmtv
Event true when OSI primitive received at Layer 4 from Layer 3 (emulate mode only).

lo_n_prmtv_avail
Indicates whether one or more OSI primitives from Layer 3 are queued and available to be received at Layer 4 (emulate mode only).

lo_n_prmtv_code
Stores code for OSI primitive received at Layer 4 in a PDU from Layer 3 (emulate mode only).

lo_n_prmtv_path
Stores path number received at Layer 4 in a PDU from Layer 3 (emulate mode only).

lo_n_sdu
Stores offset to where the SDU begins in an OSI primitive received at Layer 4 from Layer 3 (emulate mode only).

lo_p_il_buff
Stores IL buffer number received at Layer 7 in a PDU from Layer 6 (emulate mode only).

lo_p_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 7 from Layer 6 (emulate mode only).

lo_p_prmtv
Event true when OSI primitive received at Layer 7 from Layer 6 (emulate mode only).

lo_p_prmtv_avail
Indicates whether one or more OSI primitives from Layer 6 are queued and available to be received at Layer 7 (emulate mode only).

lo_p_prmtv_code
Stores code for OSI primitive received at Layer 7 in a PDU from Layer 6 (emulate mode only).

lo_p_prmtv_path
Stores path number received at Layer 7 in a PDU from Layer 6 (emulate mode only).

lo_p_sdu
Stores offset to where the SDU begins in an OSI primitive received at Layer 7 from Layer 6 (emulate mode only).

lo_ph_il_buff
Stores IL buffer number received at Layer 2 in a PDU from Layer 1 (emulate mode only).

lo_ph_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 2 from Layer 1 (emulate mode only).

lo_ph_prmtv
Event true when OSI primitive received at Layer 2 from Layer 1 (emulate mode only).

lo_ph_prmtv_avail
Indicates whether one or more OSI primitives from Layer 1 are queued and available to be received at Layer 2 (emulate mode only).

lo_ph_prmtv_code
Stores code for OSI primitive received at Layer 2 in a PDU from Layer 1 (emulate mode only).

lo_ph_prmtv_path
Stores path number received at Layer 2 in a PDU from Layer 1 (emulate mode only).

lo_ph_sdu
Stores offset to where the SDU begins in an OSI primitive received at Layer 2 from Layer 1 (emulate mode only).

lo_s_il_buff
Stores IL buffer number received at Layer 6 in a PDU from Layer 5 (emulate mode only).

lo_s_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 6 from Layer 5 (emulate mode only).

lo_s_prmtv
Event true when OSI primitive received at Layer 6 from Layer 5 (emulate mode only).

lo_s_prmtv_avail
Indicates whether one or more OSI primitives from Layer 5 are queued and available to be received at Layer 6 (emulate mode only).

lo_s_prmtv_code
Stores code for OSI primitive received at Layer 6 in a PDU from Layer 5 (emulate mode only).

lo_s_prmtv_path
Stores path number received at Layer 6 in a PDU from Layer 5 (emulate mode only).

lo_s_sdu
Stores offset to where the SDU begins in an OSI primitive received at Layer 6 from Layer 5 (emulate mode only).

lo_t_il_buff
Stores IL buffer number received at Layer 5 in a PDU from Layer 4 (emulate mode only).

lo_t_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 5 from Layer 4 (emulate mode only).

lo_t_prmtv
Event true when OSI primitive received at Layer 5 from Layer 4 (emulate mode only).

lo_t_prmtv_avail
Indicates whether one or more OSI primitives from Layer 4 are queued and available to be received at Layer 5 (emulate mode only).

lo_t_prmtv_code
Stores code for OSI primitive received at Layer 5 in a PDU from Layer 4 (emulate mode only).

lo_t_prmtv_path
Stores path number received at Layer 5 in a PDU from Layer 4 (emulate mode only).

lo_t_sdu
Stores offset to where the SDU begins in an OSI primitive received at Layer 5 from Layer 4 (emulate mode only).

load_program
Links programs together while INTERVIEW is in Run mode.

lock
Implements a lock so that processes of two or more file streams operating on a single file will not do so concurrently.

lpp_trbuff_ctl
Variables in this structure monitor the logical location in the Protocol Trace buffer.

m_a_conds
Indicates whether or not Layer 7 will respond to OSI event conditions from Layer 6.

m_bib
Stores the value of the Backward Indicator Bit for an SS#7 frame.

m_call_ref_flag
Stores the code for the call reference flag for a Q.931 packet; identifies where the call originated.

m_call_ref_len
Stores the length of the call–reference value field in a Q.931 packet.

m_cic
Store the Circuit Identification Code indicating whether MSUs are TUP or ISDN (SS#7).

m_code_type
Stores the code type for various packet headers (SS#7).

m_dl_conds
Indicates whether or not Layer 2 will respond to OSI event conditions from Layer 1.

m_fib
Stores the value of the Forward Indicator Bit for an SS#7 frame.

m_frame_addr
Stores the address of last frame detected.

m_frame_addr_cr
Stores the value of the C/R bit in a LAPD frame.

m_frame_addr_sapi
Stores the value of the Service Access Point Identifier in a LAPD frame—i.e., the network link–station address.

m_frame_addr_tei
Stores the value of the Terminal Endpoint Identifier in a LAPD frame—i.e., the address of a link station on the user side.

m_frame_bcc_type
Stores the code for the BCC type of the last frame detected.

m_frame_cntrl_byte_1
Stores the actual value of the control byte in the last frame detected.

m_frame_nr
Stores N(R) sequence number of a LAPD frame.

m_frame_ns
Stores N(S) sequence number of a LAPD frame.

m_frame_pf
Stores value of poll/final bit in last frame detected.

m_frame_type
Stores code for type of frame last detected.

m_info_element_len
Stores the length of the information element field in a Q.931 packet.

m_label_dpc
Stores the Destination Point Code (SS#7).

m_label_opc
Stores the Originating Point Code (SS#7).

m_label_sls
Stores the Signalling Link Selection (SS#7).

m_li
Stores the code for the Length Indicator byte in a frame (SS#7).

m_lo_dl_il_buff
Stores IL buffer number received at Layer 3 in a PDU from Layer 2.

m_lo_dl_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 3 from Layer 2.

m_lo_dl_prmtv
Event true when OSI primitive received at Layer 3 from Layer 2.

m_lo_dl_prmtv_avail
Indicates whether one or more OSI primitives from Layer 2 are queued and available to be received at Layer 3.

m_lo_dl_prmtv_code
Stores code for OSI primitive received at Layer 3 in a PDU from Layer 2.

m_lo_dl_prmtv_path
Stores path number received at Layer 3 in a PDU from Layer 2.

m_lo_dl_sdu_offset
Stores offset to where the Service Data Unit begins in an OSI primitive received at Layer 3 from Layer 2.

m_lo_dl_sdu_size
Stores size of Service Data Unit in an IL message buffer received at Layer 3 from Layer 2.

m_lo_n_il_buff
Stores IL buffer number received at Layer 4 in a PDU from Layer 3.

m_lo_n_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 4 from Layer 3 .

m_lo_n_prmtv
Event true when OSI primitive received at Layer 4 from Layer 3.

m_lo_n_prmtv_avail
Indicates whether one or more OSI primitives from Layer 3 are queued and available to be received at Layer 4.

m_lo_n_prmtv_code
Stores code for OSI primitive received at Layer 4 in a PDU from Layer 3.

m_lo_n_prmtv_path
Stores path number received at Layer 4 in a PDU from Layer 3.

m_lo_n_sdu_offset
Stores offset to where the Service Data Unit begins in an OSI primitive received at Layer 4 from Layer 3.

m_lo_n_sdu_size
Stores size of Service Data Unit in an IL message buffer received at Layer 4 from Layer 3.

m_lo_p_il_buff
Stores IL buffer number received at Layer 7 in a PDU from Layer 6.

m_lo_p_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 7 from Layer 6.

m_lo_p_prmtv
Event true when OSI primitive received at Layer 7 from Layer 6.

m_lo_p_prmtv_avail
Indicates whether one or more OSI primitives from Layer 6 are queued and available to be received at Layer 7.

m_lo_p_prmtv_code
Stores code for OSI primitive received at Layer 7 in a PDU from Layer 6.

m_lo_p_prmtv_path
Stores path number received at Layer 7 in a PDU from Layer 6.

m_lo_p_sdu_offset
Stores offset to where the Service Data Unit begins in an OSI primitive received at Layer 7 from Layer 6.

m_lo_p_sdu_size
Stores size of Service Data Unit in an IL message buffer received at Layer 7 from Layer 6.

m_lo_ph_il_buff
Stores IL buffer number received at Layer 2 in a PDU from Layer 1.

m_lo_ph_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 2 from Layer 1.

m_lo_ph_prmtv
Event true when OSI primitive received at Layer 2 from Layer 1.

m_lo_ph_prmtv_avail
Indicates whether one or more OSI primitives from Layer 1 are queued and available to be received at Layer 2.

m_lo_ph_prmtv_code
Stores code for OSI primitive received at Layer 2 in a PDU from Layer 1.

m_lo_ph_prmtv_path
Stores path number received at Layer 2 in a PDU from Layer 1.

m_lo_ph_sdu_offset
Stores offset to where the Service Data Unit begins in an OSI primitive received at Layer 2 from Layer 1.

m_lo_ph_sdu_size
Stores size of Service Data Unit in an IL message buffer received at Layer 2 from Layer 1.

m_lo_s_il_buff
Stores IL buffer number received at Layer 6 in a PDU from Layer 5.

m_lo_s_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 6 from Layer 5.

m_lo_s_prmtv
Event true when OSI primitive received at Layer 6 from Layer 5.

m_lo_s_prmtv_avail
Indicates whether one or more OSI primitives from Layer 5 are queued and available to be received at Layer 6.

m_lo_s_prmtv_code
Stores code for OSI primitive received at Layer 6 in a PDU from Layer 5.

m_lo_s_prmtv_path
Stores path number received at Layer 6 in a PDU from Layer 5.

m_lo_s_sdu_offset
Stores offset to where the Service Data Unit begins in an OSI primitive received at Layer 6 from Layer 5.

m_lo_s_sdu_size
Stores size of Service Data Unit in an IL message buffer received at Layer 6 from Layer 5.

m_lo_t_il_buff
Stores IL buffer number received at Layer 5 in a PDU from Layer 4.

m_lo_t_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 5 from Layer 4.

m_lo_t_prmtv
Event true when OSI primitive received at Layer 5 from Layer 4.

m_lo_t_prmtv_avail
Indicates whether one or more OSI primitives from Layer 4 are queued and available to be received at Layer 5.

m_lo_t_prmtv_code
Stores code for OSI primitive received at Layer 5 in a PDU from Layer 4.

m_lo_t_prmtv_path
Stores path number received at Layer 5 in a PDU from Layer 4.

m_lo_t_sdu_offset
Stores offset to where the Service Data Unit begins in an OSI primitive received at Layer 5 from Layer 4.

m_lo_t_sdu_size
Stores size of Service Data Unit in an IL message buffer received at Layer 5 from Layer 4.

m_message_type
Stores actual value of message–type byte in a Q.931 packet.

m_message_type_defined
Stores code for whether or not a received message type in a Q.931 packet is a defined or undefined value.

m_n_conds
Indicates whether or not Layer 3 will respond to OSI event conditions from Layer 2.

m_p_conds
Indicates whether or not Layer 6 will respond to OSI event conditions from Layer 5.

m_packet_bcc_type
Stores BCC code for a Q.931 packet.

m_packet_buff_seg
Stores IL buffer number for (X.25) packet.

m_packet_cause
Stores cause code for X.25 packet.

m_packet_d
Stores value of D–bit of last X.25 packet detected.

m_packet_daf
Stores Destination Address Field of SNA packet.

m_packet_def
Stores Destination Element Field of SNA packet.

m_packet_diag_code
Stores diagnostic code, if present, of X.25 packet.

m_packet_dsaf
Stores Destination Subarea Address Field of SNA packet.

m_packet_fi
Stores code for Format Indicator of SNA packet.

m_packet_fid_type
Stores code for FID type of SNA packet.

m_packet_info_length
Stores length of X.25 packet information, excluding the header.

m_packet_info_offset
Stores offset to where X.25 packet information begins, excluding header.

m_packet_info_ptr
Pointer to information in an X.25 packet; begins at byte immediately following packet–type byte.

m_packet_info_seg
Stores IL buffer number (X.25).

m_packet_lcn
Stores Logical Channel Number of last X.25 packet detected.

m_packet_lcn_grp
Stores Logical Channel Group Number of last X.25 packet detected.

m_packet_length
In X.25, stores length of X.25 packet information, including header. In SNA, stores length of packet, including Transmission Header and Request/Response Header.

m_packet_lsid
Stores actual value of byte for Local Session Identification in SNA packet.

m_packet_m
Stores value of M–bit of X.25 packet.

m_packet_oaf
Stores Origin Address Field of SNA packet.

m_packet_oef
Stores Origin Element Field of SNA packet.

m_packet_osaf
Stores Origin Subarea Address Field of SNA packet.

m_packet_pr
Stores P(R) sequence number of X.25 packet.

m_packet_ps
Stores P(S) sequence number of X.25 packet.

m_packet_ptr
Pointer to X.25 packet; begins at first byte in the header.

m_packet_q
Stores value of Q–bit of X.25 packet.

m_packet_rri
Stores code for Request/Response Indicator for SNA packet.

m_packet_rti
Stores code for Response Type Indicator for SNA packet.

m_packet_ru_category
Stores code for Request/Response Unit for SNA packet.

m_packet_sdi
Stores code for Sense Data Indicator for SNA packet.

m_packet_sdu_offset
Stores offset to where Service Data Unit begins in IL buffer (X.25).

m_packet_type
Stores code for packet type of last X.25 packet detected.

m_packet_type_byte
Stores actual value of packet type byte of last X.25 packet detected.

m_prot_disc
Stores actual value of protocol discriminator (Q.931).

m_ptr_to_call_ref
Pointer to call–reference value field; begins at first byte containing the call reference length (Q.931).

m_ptr_to_info_element
Pointer to information element field; begins at first byte after message–type byte (Q.931).

m_s_conds
Indicates whether or not Layer 5 will respond to OSI event conditions from Layer 4.

m_sio_ni
Stores code for Network Indicator (SS#7).

m_sio_priority
Stores priority code for U.S. format (SS#7).

m_sio_si
Stores code for Service Indicator in the Service Information Octet of MSUs (SS#7).

m_so0
Stores code for Link Status Signalling Unit types (SS#7).

m_t_conds
Indicates whether or not Layer 4 will respond to OSI event conditions from Layer 3.

m_unit_type
Stores code for signal–unit type (SS#7).

mkdir
Creates a new directory on disk.

mpm_info
Structure containing information on specific MPM board; used in *unit_config* structure.

n_conds
Indicates whether or not Layer 3 will respond to OSI event conditions from Layers 2 and/or 4 (emulate mode only).

nr_error
Event true when an N(R) error detected in received INFO or supervisory frame (emulate mode only).

ns_error
Event true when an N(S) error detected in received INFO frame (emulate mode only).

_open_space_in_il_buff
Opens up requested amount of space in IL buffer so that data may be copied into the buffer.

outsync_action
Sends one of the receivers (TD or RD) out of sync and initiates a search for sync (SYNC format only).

p_conds
Indicates whether or not Layer 6 will respond to OSI event conditions from Layers 5 and/or 7 (emulate mode only).

packet_sent
Event true when X.25 packet has been passed down to Layer 2 (emulate mode only).

pdu
Structure of an OSI primitive data unit.

ph_ch2_discard_ind
OSI primitive code for primitives moving between Layers 1 and 2. Contains the time of the run and the DCE and DTE count of frames discarded prior to the first actual frame of a new sample for the secondary port.

ph_discard_ind
OSI primitive code for primitives moving between Layers 1 and 2. Contains the time of the run and the DCE and DTE count of frames discarded prior to the first actual frame of a new sample for the main port.

ph_prmtv_type
Stores code for OSI primitives moving between Layers 1 and 2.

pos_cursor
Positions cursor in the Display Window by row and column number.

pr_error
Event true when a P(R) error is detected in X.25 DATA or supervisory packets (emulate mode only).

#pragma frame_nodes n
Allows the user to change the number of DMA command nodes on each receive line ring in Frame receive mode and SMDS receive mode.

#pragma hook
Combines compiled C code within referenced LOBJ file with compiled code of an active spreadsheet program.

#pragma il_buffer_size
Sets the size of interlayer message buffers.

#pragma il_buffers
Sets the available number of interlayer message buffers.

#pragma layer
Declares a layer.

#pragma nowarn
Suppresses warning messages from the C compiler.

#pragma object
Accesses compiled routine definitions in LOBJ file.

#pragma quad_display
Adds a softkey to enable data display of both sides of both ports when using a dual–port TIM.

#pragma sample_mode
Discards frames in High Speed Frame Mode when the Layer 1 task gets the last available IL message buffer.

#pragma set_chain_xmit

Allows multiple frames to be queued on the DMA chain.

#pragma set_high_speed_xmit

Sets the high–speed transmission capability on or off for single– and dual–port testing. Dual–port parameters for this pragma allows setting either or both ports on or off.

#pragma smds_format E1

Forces the SMDS or ATM format to E1.

#pragma smds_format DS1

Forces the SMDS or ATM format to DS1.

#pragma suppress_char_data

Suppresses capture/storage of data to dual–line, single–line, and data–plus–leads displays.

#pragma tracebuf

Configures size of data array in each user trace buffer.

prev_aux_value

Stores previous value of *curr_aux_value* variable (AUX port).

prev_date_of_day

Stores previous day of the month.

prev_display_screen

Stores previous display screen and whether unit was in Run or Freeze mode.

prev_time_of_day

Stores previous time, one minute ago.

prev_tm

Structure of previous time of day, one minute ago; instance of *tm* structure.

previous_eia_leads

Stores previous value of EIA or X.21 leads—updated only after comparison of current and previous leads made.

print_buffer

Structure of the print buffer.

_print_buffer

Instance of *print_buffer* structure. Use variables to monitor the flow of text in and out of the print buffer.

printc

Outputs a single ASCII character to the print buffer for printing.

printf

Writes output to print buffer for printing according to string that specifies how arguments should be converted for output.

prints

Writes output to print buffer under control of a string.

prog_trbuf

Structure of the Program Trace buffer.

ps_error

Event true when a P(S) error is detected in X.25 DATA packets (emulate mode only).

_raw_disk_read

Allows the user to read raw disk blocks from a floppy disk or from the hard disk.

_raw_disk_write

Allows the user to write raw disk blocks to a floppy disk or to the hard disk. *CAUTION: Using this routine will overwrite all information on your disk in the blocks selected!!! Use with care!*

rcv_buffer_full

Stores the status of the character buffer, full or not full.

rcvd_char_d_rd

Stores most recent RD character on the secondary channel—data character, BCC, flag, sync, or abort (emulate or monitor mode).

rcvd_char_d_td

Stores most recent TD character on the secondary channel—data character, BCC, flag, sync, or abort (emulate or monitor mode).

rcvd_char_rd

Stores most recent RD character—data character, BCC, flag, sync, or abort (emulate mode only).

rcvd_char_td

Stores most recent TD character—data character, BCC, flag, sync, or abort (emulate mode only).

rcvd_device_path

Stores path number connecting received X.25 packet to particular LCN and set of call parameters on X.25 Packet Level Setup screen (emulate mode only).

rcvd_frame

Event true when a frame is received (emulate mode only).

rcvd_frame_addr

Stores address of last received frame (emulate mode only).

rcvd_frame_addr_cr

Stores the value of the C/R bit of last received LAPD frame (emulate mode only).

rcvd_frame_addr_sapi

Stores the value of the Service Access Point Identifier—i.e., the network link–station address—of last received LAPD frame (emulate mode only).

rcvd_frame_addr_tei

Stores the value of the Terminal Endpoint Identifier—i.e., the address of a link station on the user side—of last received LAPD frame (emulate mode only).

rcvd_frame_bcc_type

Stores the code for BCC type of last received frame (emulate mode only).

rcvd_frame_buff_seg

Stores IL buffer number of last received frame (emulate mode only).

rcvd_frame_cntrl_byte_1

Stores actual value of control byte of last received frame (emulate mode only).

rcvd_frame_nr

Stores N(R) sequence number, if present, of last received frame (emulate mode only).

rcvd_frame_ns

Stores N(S) sequence number, if present, of last received frame (emulate mode only).

rcvd_frame_pf

Stores value of poll/final bit of last received frame (emulate mode only).

rcvd_frame_sdu_offset
Stores offset to start of Service Data Unit in IL buffer of received frame (emulate mode only).

rcvd_frame_sdu_size
Stores size of Servic Data Unit in a received frame (emulate mode only).

rcvd_frame_type
Stores code for type of frame last received (emulate mode only).

rcvd_packet
Event true when X.25 packet received from Layer 2 (emulate mode only).

rcvd_packet_ptr
Pointer to X.25 packet; begins at first byte in header of received frame (emulate mode only).

rcvd_packet_type
Stores code for X.25 packet type of last received packet (emulate mode only).

rcvd_pkt_buff_seg
Stores IL buffer number of last received X.25 packet (emulate mode only).

rcvd_pkt_cause
Stores cause code, if present, of last received X.25 packet (emulate mode only).

rcvd_pkt_d
Stores value of D–bit of last received X.25 packet (emulate mode only).

rcvd_pkt_diagn
Stores diagnostic code, if present, of last received X.25 packet (emulate mode only).

rcvd_pkt_info_length
Stores length of information in received X.25 packet, excluding header (emulate mode only).

rcvd_pkt_info_offset
Stores offset to where X.25 packet information begins, excluding the header (emulate mode only).

rcvd_pkt_info_ptr
Pointer to X.25 packet information in received packet; begins at byte immediately following packet header (emulate mode only).

rcvd_pkt_info_seg
Stores IL buffer number of received X.25 packet (emulate mode only).

rcvd_pkt_lcn
Stores Logical Channel Number of last received X.25 packet (emulate mode only).

rcvd_pkt_length
Stores length of received X.25 packet, including header information (emulate mode only).

rcvd_pkt_m
Stores value of M–bit of last received X.25 packet (emulate mode only).

rcvd_pkt_pr
Stores P(R) sequence numbering (X.25, emulate mode only).

rcvd_pkt_ps
Stores P(S) sequence numbering (X.25, emulate mode only).

rcvd_pkt_q
Stores value of Q–bit of last received packet (X.25, emulate mode only).

rcvd_pkt_sdu_offset
Stores offset to where Service Data Unit begins in IL buffer of received X.25 packet (emulate mode only).

rcvd_pkt_type_byte
Stores actual value of packet–type byte of last received X.25 packet (emulate mode only).

rd_1_resyncs
Counts the number of times that resyncing has been required on RD in HEC framing mode.

rd_1_smds_out_frame
Indicates whether the receive task is receiving valid PLCP frames (SMDS, ATM) or cell bundles (ATM); READ ONLY variable.

rd_buffer_off
Contains the offset into the il message buffer of the byte following the last DTE data byte received.

rd_il_buffer
Contains the il message buffer segment number of the buffer containing the DTE frame just received.

rd_modifier

Stores most recent modifier byte for an RD data character.

remove

Deletes named file or empty directory from disk.

rename

Renames a specified file on the active disk.

resend_frame

Sets P/F bit to specified value and resends either the first or next frame in the window (emulate mode only).

resend_frame_multi

Resends either the first or next SDLC frame in window to a specified controller address (emulate mode only).

resend_packet

Resends either the first or next X.25 packet in the window along a specified path (emulate mode only).

reset_nr

Resets to 0 the N(R) field in INFO and supervisory frames (emulate mode only).

reset_nr_multi

Resets to 0 the N(R) field in INFO and supervisory frames for a specified controller address (SDLC, emulate mode only).

reset_ns

Resets to 0 the N(S) field in INFO frames and clears the transmit window (emulate mode only).

reset_ns_multi

Resets to 0 the N(S) field in INFO frames for a specified controller address and clears the transmit window (SDLC, emulate mode only).

reset_pr_ps

Resets to 0 P(R) and P(S) fields in DATA and supervisory X.25 packets and clears the transmit window (emulate mode only).

reset_print_page

Resets INTERVIEW's automatic page numbering for printer output to 1.

restore_cursor
Returns the cursor to a previous position in the Display Window.

rewind
Returns the file-position indicator to the beginning of the file, clears the end-of-file and error indicators, and erases any memory of character returned in a previous *ungetc* operation.

rh_ptr
Pointer for Request/Response Header; begins at byte containing the RRI (SNA).

rindex
Searches for an instance of a character starting at the end of a specified list.

rmt_break
Event true when a break is received through the remote port.

rmt_flushi
Discards any characters in the remote port's input queue that have not yet been read.

rmt_flusho
Discards characters queued to be output from the remote port, but not yet transmitted.

rmt_get_baud_rate
Gets the current baud-rate setting for the remote port.

rmt_get_bits
Gets the current setting of the number of bits per character for the remote port.

rmt_get_mode
Gets the current handshaking mode for the remote port.

rmt_get_parity
Gets the current parity setting for the remote port.

rmt_getc
Reads the next character, if present, from the input queue of the remote port.

rmt_getl
Reads from the remote port one line at a time and puts input into an array.

rmt_gets
Gets specified number of characters from the remote port and puts them in an array.

rmt_input_almost_empty

Event true when the remote port's input queue transitions from more than 1/4 full to 1/4 full as data is taken out of the queue.

rmt_input_almost_full

Event true when the remote port's input queue transitions from less than 3/4 full to 3/4 full as data if being put into the queue.

rmt_input_empty

Event true when the remote port's input queue transitions from not empty to empty. All characters have been read or discarded.

rmt_input_not_empty

Event true when the remote port's input queue transitions from empty to not empty. Characters are being received.

rmt_input_overflow

Event true when the remote port's input queue transitions from not full to full. Oldest existing data in the queue is discarded to make room for new data.

rmt_lock

Locks all received characters in the remote port's input queue until they are requested.

rmt_output_empty

Event true when remote port's output queue transitions from not empty to empty. All data output to the remote port has been transmitted.

rmt_putb

Sends a string of specified length to the remote port's output queue.

rmt_putc

Sends a specified character to the remote port's output queue for transmission.

rmt_puts

Outputs a NULL–terminated string to the remote port's output queue.

rmt_resumeo

Resumes transmission of characters from the remote port (only for following handshaking mode: FDX without flow control).

rmt_send_break

Causes a break, queued as other transmits, to be transmitted from the remote port .

rmt_set_baud_rate

Sets the baud rate for the remote port.

rmt_set_bits

Sets the number of bits per character for the remote port.

rmt_set_mode

Sets the handshaking mode for the remote port.

rmt_set_parity

Set the parity for the remote port.

rmt_suspendo

Suspends transmitting if characters are queued to be output from the remote port, but have not yet been transmitted (only for following handshaking mode: FDX without flow control).

rmt_unlock

If characters are in the remote port's input queue but there are no outstanding requests for data, discards the characters.

ru_ptr

Pointer for the Request/Response Unit; begins at the first byte in the unit (SNA).

s_conds

Indicates whether or not Layer 5 will respond to OSI event conditions from Layers 4 and/or 6 (emulate mode only).

send_d_frame

Sends a specified string on the secondary channel with a user-determined BCC (emulate mode only).

send_d_frame_il

Sends a designated IL buffer out on the secondary channel (emulate mode only).

send_dl_prmtv_above

Passes a specified IL buffer from Layer 2 to Layer 3 in an OSI primitive (emulate mode only).

send_dl_prmtv_below

Passes a specified IL buffer from Layer 3 to Layer 2 in an OSI primitive (emulate mode only).

send_done_key_to_appl

Controls the function of the DONE key in Run mode.

send_frame

Adds frame-level header to IL buffer and passes buffer to Layer 1 for transmission (emulate mode only).

send_frame_structure

Structure of a frame; used in the *send_frame* routine.

send_key

Sends a specific keystroke, or sequence of keys, during Run mode as though the key(s) had been pressed. Also can change Run-mode display.

send_m_dl_prmtv_above

Passes specified IL buffer from Layer 2 to Layer 3 in an OSI monitor primitive.

send_m_n_prmtv_above

Passes specified IL buffer from Layer 3 to Layer 4 in an OSI monitor primitive.

send_m_p_prmtv_above

Passes specified IL buffer from Layer 6 to Layer 7 in an OSI monitor primitive.

send_m_ph_prmtv_above

Passes specified IL buffer from Layer 1 to Layer 2 in an OSI monitor primitive.

send_m_s_prmtv_above

Passes specified IL buffer from Layer 5 to Layer 6 in an OSI monitor primitive.

send_m_t_prmtv_above

Passes specified IL buffer from Layer 4 to Layer 5 in an OSI monitor primitive.

send_n_prmtv_above

Passes a specified IL buffer from Layer 3 to Layer 4 in an OSI primitive (emulate mode only).

send_n_prmtv_below

Passes a specified IL buffer from Layer 4 to Layer 3 in an OSI primitive (emulate mode only).

send_p_prmtv_above

Passes a specified IL buffer from Layer 6 to Layer 7 in an OSI primitive (emulate mode only).

send_p_prmtv_below
Passes a specified IL buffer from Layer 7 to Layer 6 in an OSI primitive (emulate mode only).

send_packet
Adds a packet–level header to IL buffer and passes the buffer to Layer 2 (emulate mode only).

send_packet_structure
Structure of a packet; used in the *send_packet* routine.

send_ph_prmtv_above
Passes a specified IL buffer from Layer 1 to Layer 2 in an OSI primitive (emulate mode only).

send_ph_prmtv_below
Passes a specified IL buffer from Layer 2 to Layer 1 in an OSI primitive (emulate mode only).

send_s_prmtv_above
Passes a specified IL buffer from Layer 5 to Layer 6 in an OSI primitive (emulate mode only).

send_s_prmtv_below
Passes a specified IL buffer from Layer 6 to Layer 5 in an OSI primitive (emulate mode only).

send_stat_message
Sends stat message structure and addresses of counter, timers, and accumulators to the 68010 CPU (CPM board).

send_t_prmtv_above
Passes a specified IL buffer from Layer 4 to Layer 5 in an OSI primitive (emulate mode only).

send_t_prmtv_below
Passes a specified IL buffer from Layer 5 to Layer 4 in an OSI primitive (emulate mode only).

set_a_conds
Determines whether or not Layer 7 will respond to OSI event conditions from Layer 6; use it to write to variable *a_conds* (emulate mode only).

set_aux_ctl_leads
Determines whether or not AUX port leads will be control leads. Control leads must also be input leads.

set_aux_direction
Designates leads on the AUX port as input or output.

set_aux_reg
Reconfigures the CIO chip (AUX port).

set_cpm_date_time
Sets INTERVIEW's real–time clock.

set_dl_conds
Determines whether or not Layer 2 will respond to OSI event conditions from Layers 1 and/or 3; use it to write to variable *dl_conds* (emulate mode only).

set_dw_fkey_label
Assigns a user–defined label to a specified Display Window softkey.

_set_file_type
Sets the file–type label (on the File Maintenance screen) for a specified file.

_set_hex_led
Toggles the LED on the HEX key on and off.

set_isdn_speaker_chan
Allows programmer to control the speaker located on the ISDN mux board.

set_ltrace_fkey_label
Modifies specified Protocol Trace buffer softkey label.

set_m_a_conds
Determines whether or not Layer 7 will respond to (OSI) monitor event conditions from Layer 6; use it to write to variable *m_a_conds*.

set_m_dl_conds
Determines whether or not Layer 2 will respond to (OSI) monitor event conditions from Layer 1; use it to write to variable *m_dl_conds*.

set_m_n_conds
Determines whether or not Layer 3 will respond to (OSI) monitor event conditions from Layer 2; use it to write to variable *m_n_conds*.

set_m_p_conds
Determines whether or not Layer 6 will respond to (OSI) monitor event conditions from Layer 5; use it to write to variable *m_p_conds*.

set_m_s_conds
Determines whether or not Layer 5 will respond to (OSI) monitor event conditions from Layer 4; use it to write to variable *m_s_conds*.

set_m_t_conds
Determines whether or not Layer 4 will respond to (OSI) monitor event conditions from Layer 3; use it to write to variable *m_t_conds*.

_set_maint_buff_bit
Sets a new maintain bit for a specified IL buffer to: 1) hold the buffer at a particular layer, or 2) pass the buffer to another layer.

set_max_frame_length
Sets the frame length at which each subsequently received frame will be truncated. (May be delayed for up to five frames from each side of the line.)

set_n_conds
Determines whether or not Layer 3 will respond to OSI event conditions from Layers 2 and/or 4; use it to write to variable *n_conds* (emulate mode only).

set_p_conds
Determines whether or not Layer 6 will respond to OSI event conditions from Layers 5 and/or 7; use it to write to variable *p_conds* (emulate mode only).

set_print_header
Writes output to print buffer to be printed after each form feed.

set_s_conds
Determines whether or not Layer 5 will respond to OSI event conditions from Layers 4 and/or 6; use it to write to variable *s_conds* (emulate mode only).

set_t_conds
Determines whether or not Layer 4 will respond to OSI event conditions from Layers 3 and/or 5; use it to write to variable *t_conds* (emulate mode only).

set_tcr_b
Clamps the transmit line to 0 (space) or 1 (mark), or unclamps it so that X.21 transmit routines may be executed. In non–X.21 environment, initiates and terminates a break.

set_tim_setup
Dynamically changes the B channel on which emulation or monitoring is being done. Use with *get_tim_setup*.

set_utrace_fkey_label
Modifies specified user trace softkey label.

show_dw_fkey_labels
Updates display of all user–assigned softkey labels in the Display Window.

signal
Conveys instructions to other tests and layers where conditions are monitoring the signal by name.

*signal_*name
Event true when named signal is detected (the argument of *signal* routine).

slip_bit
Slips one DTE data bit each time it is invoked to obtain byte synchronization.

slip_dce_bit
Slips one data bit each time it is invoked to obtain byte synchronization.

slip_dte_bit
Slips one data bit each time it is invoked to obtain byte synchronization on the secondary derived clock.

smds_format
Determines the SMDS format that is in effect, E1 or DS1; READ ONLY variable.

smds_il_buffer
Structure of an SMDS interlayer–message buffer, both directions.

sound_alarm
Sounds INTERVIEW's alarm.

sprintf
Writes output to a character array (attributes not supported) according to a string that specifies how the arguments should be converted for output.

ss7_compression_mode
Allows suppression or compression of FISUs (Fill–In Signal Units), exactly as in the SS#7 compression package.

_start_il_buff_list
Starts a linked list of text inside an IL buffer.

start_of_run_date
Date when Run mode was entered; used to timestamp events.

start_of_run_time

Time when Run mode was entered; used to timestamp events.

start_rcrd_play

Depending on Line Setup configuration, activates either data recording or playback.

stat_msg

Structure of stat message sent for counters, timers, or accumulators.

stracef

Writes output to a variable according to a string that specifies how arguments should be converted for output.

strcmp

Matches a user–entered string to line data.

surrender_cpu

Surrenders the CPU, placing calling task onto the end of the ready queue. *CAUTION: DO NOT USE THIS ROUTINE AT LAYER 1; to do so will put the INTERVIEW into an endless loop.*

suspend_rcrd_play

Depending on Line Setup configuration, suspends either data recording or playback.

t_conds

Indicates whether or not Layer 4 will respond to OSI event conditions from Layers 3 and/or 5 (emulate mode only).

td_1_resyncs

Counts the number of times that resyncing has been required on TD in HEC framing mode.

td_1_smds_out_frame

Indicates whether the receive task is receiving valid PLCP frames (SMDS, ATM) or cell bundles (ATM); READ ONLY variable.

td_buffer_off

Contains the offset into the il message buffer of the byte following the last DTE data byte received.

td_il_buffer

Contains the il message buffer segment number of the buffer containing the DTE frame just received.

td_modifier
Stores most recent modifier byte for a TD data character.

temporary_prompt
Blanks the prompt line, displays a new user–defined prompt, and then waits a user–defined interval before blanking the prompt line again.

th_ptr
Pointer for the Transmission Header; begins at the byte containing FID type (SNA).

timeout
Structure of a timeout.

*timeout*_name_*expired*
Event signaled by program as part of *timeout_restart_action* routine.

timeout_restart_action
Starts a named timeout timer running down from a specified value. When the timer reaches 0, a named function is called.

timeout_stop_action
Stops a named timeout timer, preventing it from expiring.

timer_struct
Structure of a timer.

timestamp_struct
Structure of a timeout.

tm
Structure of time of day.

trace_buf
Structure of a trace buffer.

trace_buffer_header
Structure of header for a trace buffer.

tracec
Outputs a single ASCII character to a specified trace screen.

tracef
Writes output, with or without attributes, to a specified trace screen according to a string that specifies how arguments should be converted for output.

traces

Writes output to a specified trace screen under control of a string.

ungetc

Temporarily forces a specified character into a variable associated with an input stream, overwriting the previous *ungetc* character.

unhighlight_dw_fkey_label

Displays a specified user–defined softkey label in the Display Window in normal video.

unit_config

Structure containing a unit's hardware and software configuration.

unit_setup

Structure containing Line Setup and FEB tick–rate selections.

unlock

Inverse of lock routine. Sleeping tasks woken up to try to claim the lock. One will succeed, the rest go back to sleep.

up_dl_il_buff

Stores IL buffer number received at Layer 2 in a PDU from Layer 3 (emulate mode only).

up_dl_pdu_seg

Stores iAPX–286 segment number for PDU received at Layer 2 from Layer 3 (emulate mode only).

up_dl_prmtv

Event true when an OSI primitive is received at Layer 2 from Layer 3 (emulate mode only).

up_dl_prmtv_avail

Indicates whether one or more OSI primitives from Layer 3 are queued and available to be received at Layer 2 (emulate mode only).

up_dl_prmtv_code

Stores OSI primitive code received at Layer 2 in a PDU from Layer 3 (emulate mode only).

up_dl_prmtv_path

Stores path number received at Layer 2 in a PDU from Layer 3 (emulate mode only).

up_dl_sdu
Stores offset to the start of the Service Data Unit in an IL buffer received at Layer 2 from Layer 3 (emulate mode only).

up_n_il_buff
Stores IL buffer number received at Layer 3 in a PDU from Layer 4 (emulate mode only).

up_n_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 3 from Layer 4 (emulate mode only).

up_n_prmtv
Event true when an OSI primitive is received at Layer 3 from Layer 4 (emulate mode only).

up_n_prmtv_avail
Indicates whether one or more OSI primitives from Layer 4 are queued and available to be received at Layer 3 (emulate mode only).

up_n_prmtv_code
Stores OSI primitive code received at Layer 3 in a PDU from Layer 4 (emulate mode only).

up_n_prmtv_path
Stores path number received at Layer 3 in a PDU from Layer 4 (emulate mode only).

up_n_sdu
Stores offset to the start of the Service Data Unit in an IL buffer received at Layer 3 from Layer 4 (emulate mode only).

up_p_il_buff
Stores IL buffer number received at Layer 6 in a PDU from Layer 7 (emulate mode only).

up_p_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 6 from Layer 7 (emulate mode only).

up_p_prmtv
Event true when an OSI primitive is received at Layer 6 from Layer 7 (emulate mode only).

up_p_prmtv_avail
Indicates whether one or more OSI primitives from Layer 7 are queued and available to be received at Layer 6 (emulate mode only).

up_p_prmtv_code
Stores OSI primitive code received at Layer 6 in a PDU from Layer 7 (emulate mode only).

up_p_prmtv_path
Stores path number received at Layer 6 in a PDU from Layer 7 (emulate mode only).

up_p_sdu
Stores offset to the start of the Service Data Unit in an IL buffer received at Layer 6 from Layer 7 (emulate mode only).

up_ph_prmtv_avail
Indicates whether one or more OSI primitives from Layer 2 are queued and available to be received at Layer 1 (emulate mode only).

up_s_il_buff
Stores IL buffer number received at Layer 5 in a PDU from Layer 6 (emulate mode only).

up_s_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 5 from Layer 6 (emulate mode only).

up_s_prmtv
Event true when an OSI primitive is received at Layer 5 from Layer 6 (emulate mode only).

up_s_prmtv_avail
Indicates whether one or more OSI primitives from Layer 6 are queued and available to be received at Layer 5 (emulate mode only).

up_s_prmtv_code
Stores OSI primitive code received at Layer 5 in a PDU from Layer 6 (emulate mode only).

up_s_prmtv_path
Stores path number received at Layer 5 in a PDU from Layer 6 (emulate mode only).

up_s_sdu
Stores offset to the start of the Service Data Unit in an IL buffer received at Layer 5 from Layer 6 (emulate mode only).

up_t_il_buff
Stores IL buffer number received at Layer 4 in a PDU from Layer 5 (emulate mode only).

up_t_pdu_seg
Stores iAPX–286 segment number for PDU received at Layer 4 from Layer 5 (emulate mode only).

up_t_prmtv
Event true when an OSI primitive is received at Layer 4 from Layer 5 (emulate mode only).

up_t_prmtv_avail
Indicates whether one or more OSI primitives from Layer 5 are queued and available to be received at Layer 4 (emulate mode only).

up_t_prmtv_code
Stores OSI primitive code received at Layer 4 in a PDU from Layer 5 (emulate mode only).

up_t_prmtv_path
Stores path number received at Layer 4 in a PDU from Layer 5 (emulate mode only).

up_t_sdu
Stores offset to the start of the Service Data Unit in an IL buffer received at Layer 4 from Layer 5 (emulate mode only).

user_setup_struct
Structure of user setup for ISDN S/T/U including routine version, revision of TIM and MUX, and data and voice channels.

window_color
Stores the current color attribute (low byte) and enhancements (high byte) of data words in the Display Window.

window_modifier
Stores current font modifiers for the Display Window.

write_aux
Sends a combination of data, control, and (perhaps) unused bits as output. Input bits not transmitted (AUX port).

x21_idle_action
Changes the idle–line conditions applied to SYNC data by the INTERVIEW (X.21).

x21_transmit_call
Sends a specified data string in X.21 call–setup mode.

x21_transmit_call_idle

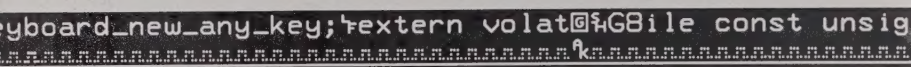
Sends a data string in x.21 call–setup mode and includes a specified idle character.

xmit_list

Structure of a transmit list for *l1_transmit*, *x21_transmit_call*, and *send_d_frame* routines.

xtim_load_program

Allows the user to change the TIM selection within the TIM Expansion Shelf and run the same program or a different program. The unit automatically loads the new program, the new TIM, and recompiles as necessary.



GENERAL SIGNAL
Networks

13000 Midlantic Drive, Mount Laurel, New Jersey 08054